ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACIÓN

UNIVERSIDAD DE CANTABRIA

Trabajo Fin de Máster

Sensores con inteligencia artificial
(Sensors with artificial intelligence)

Para acceder al Titulo de

Máster Universitario en Ingeniería de Telecomunicación

Autor: Iulian Antonov

09 - 2021

E.T.S. DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACION

MASTER UNIVERSITARIO EN INGENIERÍA DE TELECOMUNICACIÓN

CALIFICACIÓN DEL TRABAJO FIN DE MASTER

Realizado por: Iulian Antonov
Director del TFM: Pablo Pedro Sánchez Espeso
Título: "Sensores con inteligencia artificial"
Title: "Sensors with artificial intelligence"

Presentado a examen el día: 29\09\2021

para acceder al Título de

MASTER UNIVERSITARIO EN INGENIERÍA DE TELECOMUNICACIÓN

Composición del Tribunal:
Presidente (Apellidos, Nombre): José Luis Arce Diego
Secretario (Apellidos, Nombre): Victor Fernández Solórzano
Vocal (Apellidos, Nombre): Hector Posadas Cobo

Este Tribunal ha resuelto otorgar la calificación de: .....

Fdo.: El Presidente

Fdo.: El Secretario

Fdo.: El Vocal

Fdo.: El Director del TFM
(sólo si es distinto del Secretario)

Vº Bº del Subdirector

Trabajo Fin de Máster Nº
(a asignar por Secretaría)

Agradecimientos

En primer lugar quiero agradecer a mi tutor Pablo Pedro Sánchez Espeso, quien con sus conocimientos y apoyo me guió a través de cada una de las etapas de este trabajo para alcanzar los resultados que buscaba.

También quiero agradecer al Centro de Investigación y Formación Agrarias y a la Universidad de Cantabria por brindarme todos los recursos y herramientas que fueron necesarios para llevar a cabo el proceso de investigación. No hubiese podido arribar a estos resultados de no haber sido por su incondicional ayuda.

Por último, quiero agradecer a todos mis compañeros y a mi familia, por apoyarme aún cuando mis ánimos decaían. En especial, quiero hacer mención de mis padres, que siempre estuvieron ahí para darme palabras de apoyo y un abrazo reconfortante para renovar energías.

Muchas gracias a todos.

Contents

List of Figures

Chapter 1

Motivación

En la industria moderna, la productividad, la calidad, la fiabilidad y la seguridad dependen en gran medida del rendimiento de los sensores empleados. Forman una interfaz entre los equipos de producción y el entorno que los rodea, proporcionando información basada en los resultados de las operaciones ejecutadas. Por lo tanto, los sensores pueden encontrarse en una gama extremadamente amplia de aplicaciones en los sistemas industriales, en los que desempeñan un papel muy importante. El primer elemento de cualquier sistema de control y medición es el propio sensor. El rendimiento del sensor define el rendimiento del sistema de control/medición y el del sistema industrial en su conjunto. No es posible distinguir entre la información correcta y la incorrecta proporcionada por un sensor, a menos que se utilice información adicional proporcionada por otro sensor. Esto valida la afirmación: Ninguna máquina puede funcionar mejor que sus sensores.

Un rasgo característico de los sensores en las aplicaciones industriales son las condiciones de trabajo extremas. A menudo, los sensores utilizados en la industria tienen que ofrecer un rendimiento excelente en entornos difíciles e inaccesibles (por ejemplo, temperatura muy alta o muy baja, alta humedad, vacío, tratamiento agresivo, vibraciones, interferencias, espacio limitado, consumo de energía reducido, etc.) sin posibilidad de calibración periódica. La sustitución de estos sensores puede ser muy compleja, requerir mucho tiempo y, por tanto, ser muy cara, incluso cuando el propio sensor es de bajo coste. Por eso, además de cumplir su función principal, los sensores utilizados en la industria tienen que poseer características de funcionalidad adicionales, como el autodiagnóstico, la auto calibración, el funcionamiento autónomo con un consumo mínimo de energía, la compatibilidad con redes de sensores por cable o inalámbricas (WSN) y un factor de forma pequeño. Además, junto con la mayor funcionalidad, los sensores industriales deben ser extremadamente robustos y fiables. Para cumplir todos los requisitos mencionados, estos sensores deben poseer un cierto nivel de inteligencia.

Teniendo en cuenta el impacto y la importancia que tienen los sensores en la actualidad para la industria, este trabajo se enfocará en el uso efectivo de éstos mediante la aplicación de inteligencia artificial. Un dispositivo con varios sensores capaz de aprender y ser programado ante casi cualquier experimento o implementación industrial, se convierte en una herramienta muy potente para la automatización de procesos o el aumento de la producción en un sector industrial en concreto. Una herramienta como la mencionada se podría utilizar para la recolección de datos y generación de árboles de decisión que le aportarían al dispositivo de inteligencia

artificial. La capacidad del dispositivo para la detección de estados, patrones de movimiento, sonido, temperatura y demás valores físicos lo convierte en una alternativa ideal para algunos sistemas de supervisión. Podría utilizarse como una adición a un dispositivo más complejo para la gestión de estados. En definitiva, una herramienta muy versátil para usos complejos. El artículo [8] habla sobre el bienestar animal y los beneficios que implica. Teniendo en cuenta esta información utilizar un dispositivo capaz de detectar los movimientos de los animales, saber con exactitud qué comen y en general detectar el bienestar de un animal podría tener un impacto muy positivo en la producción ganadera o en la detección del estado de una mascota.

Este trabajo está enfocado en demostrar la posibilidad de uso de un dispositivo con varios sensores y hardware de inteligencia artificial para la implementación en la automatización y aumento de productividad. Teniendo en cuenta que el dispositivo utilizado dispone de varios sensores capaces de detectar una variedad amplia de magnitudes físicas, se pretende recolectar datos con el dispositivo para entrenarlo posteriormente a distinguir y detectar patrones de datos que podrían asociarse a estados distintos del objeto al que está anclado el dispositivo, como por ejemplo el estilo de conducción de una persona donde se podría detectar la eficiencia en el estilo de conducción o analizar el estado del coche mediante las vibraciones que produce el mismo en la carretera, en el caso de que la presión de las ruedas no sea la indicada, entre otras cosas.

Keywords – Sensors, artificial intelligence, data analysis.

Chapter 2

Estado del arte

2.1 Análisis de datos

El análisis de datos es el proceso de inspección, limpieza, transformación y modelado de datos con el objetivo de descubrir información útil, sacar conclusiones y apoyar la toma de decisiones [11]. El análisis de datos tiene múltiples facetas y enfoques, que engloban diversas técnicas bajo una variedad de nombres, y se utiliza en diferentes ámbitos de la empresa, la ciencia y las ciencias sociales [29]. En el mundo empresarial actual, el análisis de datos desempeña un papel importante a la hora de tomar decisiones más científicas y de ayudar a las empresas a operar con mayor eficacia [33].

2.1.1 Descubrimiento de datos

Cada proyecto de investigación debe empezar por la búsqueda de datos existentes que sean relevantes para el tema de investigación. Esto es importante para los proyectos basados en el análisis secundario (que reutilizan datos producidos por otro proyecto de investigación), pero también es importante para los proyectos que pretenden recoger datos originales [35].

El descubrimiento de datos es un proceso de varios pasos distintos -y cíclicos-. En el proceso de descubrimiento de datos, es importante ser consciente del tipo de datos que se están buscando. ¿Qué datos se ajustan a las intenciones de investigación?

Enumerar las características de los datos que se quieren descubrir facilita:

1. Formular los términos de búsqueda adecuados para encontrar fuentes que contengan esos datos.

2. Buscar en la fuente de datos elegida los datos adecuados.

Antes de empezar a buscar datos, hay que estar seguro del tema o dominio que nos interesa. Puede ser la política, la salud, la familia, las desigualdades sociales, etc. A continuación, debe estar seguro de sus intenciones de investigación. ¿Cuál es su pregunta de investigación? Una pregunta de investigación es una o varias preguntas a las que su estudio quiere dar respuesta.

Su pregunta de investigación contiene varios conceptos, es decir, conceptos científicos desarrollados para la investigación sistemática de la cuestión. Ejemplos de estos conceptos son "empleo", "edad" o "educación". Cuando busques datos adecuados para

tu investigación, busca indicadores de esos conceptos. En la investigación con encuestas, estos indicadores son las variables contenidas en el conjunto de datos. El concepto de "educación" puede tener varios indicadores, siendo los más comunes "la educación más alta completada" medida en categorías (estandarizadas), o "años de escolarización" medidos en el total de años pasados por el encuestado en las escuelas [35].

También existen conceptos complejos que utilizan información de más de una pregunta/indicador de la encuesta. Por ejemplo, la participación política es un concepto multidimensional que incluye el voto, la pertenencia a una organización y la manifestación, etc. Una vez definidos los conceptos con los que se quiere trabajar, es necesario definir qué indicadores de los conceptos se necesitan encontrar en los datos. ¿Tiene un nivel de medición preferido para sus variables clave, es decir, busca variables medidas a nivel nominal, ordinal o de intervalo?

La investigación debe seguir una teoría previamente desarrollada. Es necesario buscar conceptos y los indicadores que hayan sido utilizados previamente por otros investigadores.

Tras definir el estudio que se va a realizar, el investigador debe definir las características específicas que deben tener los datos. Por ejemplo, la población que se pretende estudiar, el ámbito temporal deseado, los valores geográficos, los conjuntos de datos de calidad frente a los de cantidad y las condiciones previas que deben establecerse si las hay.

Una vez se encuentre un recurso que albergue el tipo de datos de interés, se deberá averiguar cómo buscar en el archivo o repositorio de datos elegido. Para traducir las necesidades en una solicitud de búsqueda, se tendrá que averiguar qué funcionalidades de búsqueda ofrece el recurso de datos. Dichas funcionalidades de búsqueda difieren para cada sistema de búsqueda individual. Al buscar datos, se pueden obtener demasiados (en su mayoría irrelevantes), muy pocos o ningún resultado. Suponiendo que los datos puedan encontrarse en la fuente de datos elegida, se puede intentar formular una consulta de búsqueda ampliando o reduciendo el alcance (por ejemplo, utilizando menos términos de búsqueda en el campo de búsqueda o siendo más restrictivo mediante el uso de filtros).

Cuando se recolectan datos, se deberá decidir si estos son relevantes para la investigación. Si los datos no parecen relevantes, se tendría que volver a los pasos anteriores del ciclo de descubrimiento de datos si es necesario y ajustar la estrategia de búsqueda. Por último, para determinar la calidad de los datos, hay que familiarizarse con su contenido y hacerse una idea detallada de lo que contienen y lo que no [35].

2.1.2 Minería de datos

El uso de los ordenadores ha proporcionado una enorme cantidad de datos a nuestra disposición. Debido a la creciente cantidad de datos, los expertos se han enfrentado a retos a la hora de extraer información útil. Esto ha dado lugar a la minería de datos.

La minería de datos es un proceso no trivial de extracción de información oculta, previamente desconocida y potencialmente útil, a partir de grandes bases de datos. La minería de datos también puede explicarse como la búsqueda de las correlaciones en una gran base de datos relacional a partir de los diferentes ángulos de profundidad

con los que la analizamos. Es una poderosa herramienta con gran potencial que ayuda a las organizaciones o empresas a obtener más beneficios de la información recogida [7].

La minería de datos nos proporciona la información útil que las consultas y los informes no son capaces de proporcionarnos de manera eficiente. La información que se extrae mediante la etiqueta de minería de datos no está explícitamente disponible en la base de datos, mientras que la aplicación de base de datos sólo proyecta la información que está disponible en el banco de información con una capacidad de manipulación restringida. Por lo tanto, la minería de datos se describe mejor como el descubrimiento de conocimientos en las bases de datos.

La minería de datos se compone de siete fases, las cuatro primeras se utilizan para el preprocesado de datos, es decir, los datos se preparan en un formato para su uso posterior y las tres restantes se utilizan para trabajar en los datos de forma que se recupera la información oculta y/o se clasifica dicha información.

La limpieza de datos se utiliza para eliminar todo el ruido y otros datos incoherentes de la base de datos. La integración de datos se utiliza para adaptar estos para un propósito concreto, ya que éstos pueden proceder de diversas fuentes. El almacén de datos es un lugar en el que se guardan todos los datos limpios e integrados. La fase de selección de datos filtra los datos más adecuados para la tarea de minería de datos. La transformación de los datos los convierte en un formato adecuado para la minería de datos. La fase de minería de datos utiliza métodos inteligentes para generar el conocimiento o los patrones. Estos patrones se evalúan en la fase siguiente, que es la fase de evaluación de los patrones, y en la última fase el conocimiento se presenta en un formato fácil de usar.

La base de datos o el almacén de datos se utiliza para introducir los datos en bruto; puede haber muchas incoherencias en ellos, por lo que hay que limpiarlos e integrarlos en este componente del sistema. El segundo componente principal del sistema es basado en el conocimiento. Este componente se utiliza para aplicar diferentes técnicas como la clasificación, la agrupación, etc., sobre los datos y generar patrones para la evaluación. El siguiente componente es la evaluación de patrones, se utiliza para diferenciar entre los patrones de interés y los no relevantes. El último componente es la interfaz gráfica de usuario, que representa los patrones erradicados en diferentes formas.

La minería de datos puede aplicarse en bases de datos relacionales o bases de datos transaccionales para descubrir patrones con éxito. Las bases de datos temporales son aquellas cuyos datos se modulan a lo largo del tiempo, por ejemplo: las bases de datos de los bancos, pueden ser minadas de forma conjunta. La minería de datos también puede aplicarse en las bases de datos de texto que albergan palabras, por ejemplo, dar un breve resumen de un largo dato de texto de entrada [7]. La World Wide Web e Internet también hacen uso de técnicas de minería para averiguar los patrones de comportamiento de los usuarios. Esta técnica se denomina minería de patrones de recorrido. Es efectiva para tomar decisiones de marketing, como encontrar las páginas más visitadas y colocar publicidad en ellas.

Las técnicas de minería de datos pueden clasificarse a grandes rasgos en dos categorías: descriptivas y predictivas [36]. La descriptiva es la técnica que nos informa de todas las propiedades de los datos de entrada. La predictiva es el tipo de técnica que realiza inferencias en los datos de entrada para generar o predecir la información que está oculta. La distinción entre descripción y predicción no es muy

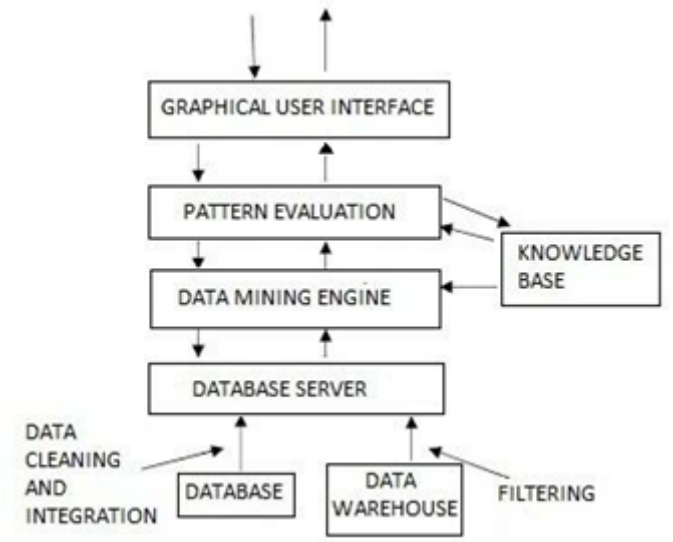


Figure 2.1: Arquitectura de la minería de datos [7]

clara. Los modelos de predicción también pueden ser descriptivos (en la medida en que sean comprensibles), y los modelos descriptivos pueden ser utilizados para la predicción. Para lograr estos objetivos, las categorías de predicción y de descripción se asocian a las cinco operaciones básicas, tal y como se presentan en la 2.2.

Aunque sólo hay un par de operaciones básicas de minería de datos, existe una gran variedad de técnicas de minería de datos que hacen posible estas operaciones. Los sistemas de minería de datos no suelen incluir cada una de estas técnicas, sino que suelen combinar dos o más técnicas diferentes entre las que el usuario/ingeniero puede elegir, en función del problema concreto. Por lo tanto, los usuarios potenciales deberían estudiar las técnicas más comunes, para decidir cuál se ajusta mejor a sus necesidades de ingeniería. La 2.3 presenta algunas técnicas comunes asignadas a las operaciones básicas de minería de datos, haciendo hincapié en los problemas de clasificación y regresión.

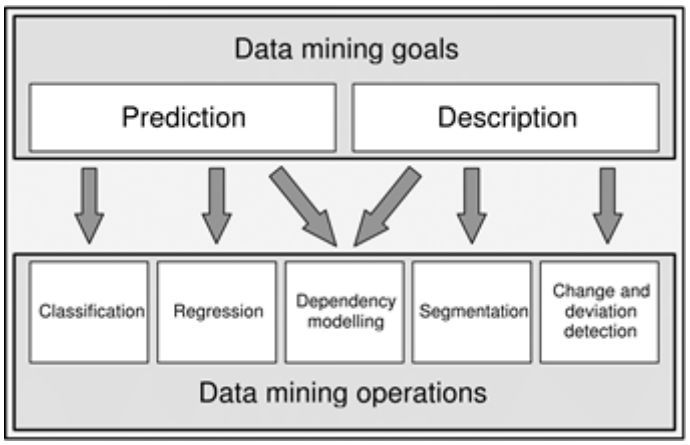


Figure 2.2: La conexión entre los objetivos de la minería de datos y las operaciones [36]

Los dos tipos centrales de problemas de predicción en ingeniería son la clasificación y la regresión. Se examinan muestras/observables de experiencias pasadas

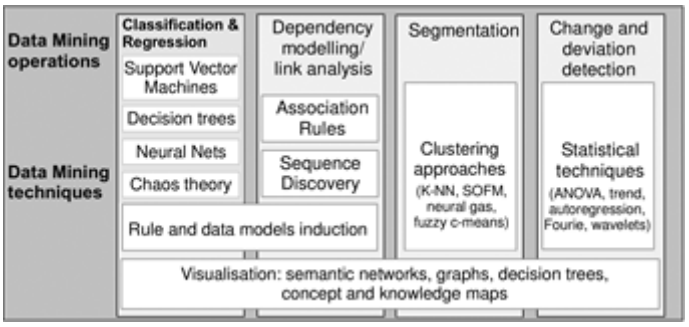


Figure 2.3: Operaciones y técnicas de análisis de datos [36]

con atributos conocidos (características) y se generalizan a casos futuros. La clasificación está estrechamente vinculada a la agrupación, que consiste en identificar los clusters incrustados en el espacio de datos multidimensional, donde un cluster es una colección de objetos de datos (grupos de datos) que son "similares" entre sí.

La similitud suele expresarse como diferentes funciones de distancia. En la literatura se han propuesto varios enfoques para desarrollar clasificadores mediante clustering, que pueden resumirse como:

- Clustering iterativo
- Clustering jerárquico aglomerativo
- Clustering jerárquico divisivo [36]

El problema de la regresión es muy similar al de la clasificación. Suele describirse como un proceso de inducción del modelo de datos del sistema que será capaz de predecir respuestas del sistema que aún no se han observado. En el caso de la regresión, la respuesta del sistema suele ser un valor real, mientras que en la clasificación es la etiqueta o etiquetas de clase. La predicción de series temporales es un tipo especializado de problema de regresión (u ocasionalmente de clasificación), en el que se toman medidas/observables a lo largo del tiempo para las mismas características. Desde el punto de vista de la minería de datos predictiva, los datos de series temporales aumentan enormemente las dimensiones de la resolución de problemas en una dirección completamente diferente. En lugar de casos con un valor medido para cada característica, los casos tienen la misma característica medida en diferentes momentos. Para superar este problema, los datos brutos dependientes del tiempo suelen transformarse para la minería de datos predictiva en un espacio de datos de menor dimensión utilizando transformaciones como la cuantificación vectorial y los métodos de espacio de estados o se aplican simples métodos de promediación y remuestreo [36].

Clasificación de datos

La clasificación de datos es el proceso de formación de un modelo o clasificador para predecir las etiquetas categóricas, es decir, las etiquetas de las clases de datos distinguidas. Estos modelos así creados se utilizan para delinear las etiquetas de los nuevos datos de entrada o los datos cuyas etiquetas de clase no se conocen. La clasificación de datos es un proceso de dos pasos [7].

- En el primer paso se crea la función o el modelo que dilucida el conjunto de clases de datos basándose en la tupla de la base de datos ya presentada y sus etiquetas de clase concordantes. Esta fase suele denominarse fase de entrenamiento porque el modelo se basa en las características de los datos de entrenamiento.

- En el segundo paso se comprueba la precisión o consistencia del esqueleto así creado. Es decir, se crean algunas tuplas de prueba y se aplican al modelo si se clasifican correctamente, entonces el modelo creado es preciso y puede utilizarse para clasificar los datos cuyas etiquetas no se conocen. Es necesario crear estas tuplas de prueba porque si la tupla de entrenamiento se cumple, entonces se ajustará correctamente al modelo.

El árbol de decisión, reglas de clasificación o reglas "if then", fórmulas matemáticas, redes neuronales, etc. se utilizan habitualmente para emblematizar el modelo de clasificación. En un árbol de decisión cada nodo representa el valor del atributo, la rama denota el resultado de la prueba y las hojas denotan las clases de datos. Por ejemplo, la 2.4 en la que la perspectiva de la población, altura, color de pelo y ojos son los atributos, las clases uno ó dos son la clasificación de los datos.

Index ofexample	Height	Color of hair	Color of eyes	Class
1	Low	Blond	Bleu	1
2	Low	Brown	Bleu	1
3	Tall	Brown	Hazel	1
4	Tall	Blond	Hazel	2
5	Tall	Brown	Bleu	2
6	Low	Blond	Hazel	2
7	Tall	Red	Bleu	2
8	Tall	Blond	Bleu	2

Figure 2.4: Ejemplo modelo de clasificación [27].

Clasificador Bayesiano

La clasificación Bayesiana es un enfoque de la clasificación no supervisada basado en el modelo de mezcla clásico [36], complementado con un método Bayesiano para determinar las clases óptimas. En el enfoque Bayesiano de la clasificación no supervisada, el objetivo es encontrar el conjunto más probable de descripción de clases (un clasificador) dados los datos y las expectativas previas. La introducción de las expectativas previas impone automáticamente un equilibrio entre el ajuste a los datos y la complejidad de las descripciones de las clases. No existe una forma generalmente aceptada de valorar la calidad relativa de las clasificaciones alternativas. Los métodos de elaboración de modelos y de búsqueda de conjuntos de clases descriptivas han sido objeto de investigación estadística durante muchos años. La mayoría de los clasificadores Bayesianos utilizan un modelo que da la probabilidad de los datos condicionada al modelo hipotetizado: P(X—H, p) , conocida como función de verosimilitud. La estimación de máxima verosimilitud (MLE) se ocupa de encontrar el conjunto de modelos y parámetros que maximiza esta probabilidad. Sin embargo, la MLE no suele proporcionar una forma convincente de comparar clasificaciones alternativas que difieren en los modelos de clase y/o en el número de clases. La MLE suele aumentar con la complejidad del modelo y el número de clases

(hasta que el número de clases es igual al número de casos). El enfoque alternativo consiste en averiguar la probabilidad de diferentes modelos hipotetizados (modelos probabilísticos) dados los datos, P(H—X) y luego comparar los modelos, que en este caso tienen diferente número de clases. Esta estrategia se emplea en el algoritmo de clasificación Bayesiano AutoClass [14].

Inducción de árboles de decisión a partir de datos

Un árbol de decisión simple es una herramienta de clasificación que utiliza una estructura de grafos en forma de árbol. El vector de características se divide en regiones únicas, correspondientes a las clases, de forma secuencial [10]. Al presentar un vector de características, la región a la que se asignará el vector de características se busca a través de una secuencia de decisiones a lo largo de un camino de nodos de un árbol adecuadamente construido. Dado un vector de características de entrada $X \in R^n$ se construye un árbol de decisión binario con los siguientes pasos.

Para las consultas categóricas se formula un conjunto de preguntas binarias (verdadero/falso), en las que se establece un valor de umbral adecuado. Para cada característica, cada valor posible del umbral 'Cj' define una división específica del subconjunto 'X'. Cada división binaria de un nodo genera dos nodos descendientes. Un criterio de división del árbol 't' se basa en una función de impureza del nodo 'I(t)'.

Se puede adaptar una regla simple para detener la división, cuando el valor máximo de DI(t), sobre todas las divisiones posibles, es menor que un umbral T; entonces, se detiene la división. Otras alternativas son dejar de dividir cuando la cardinalidad del subconjunto 'Xt' es lo suficientemente pequeña o cuando 'Xt' es puro, en el sentido de que todos los puntos en él pertenecen a una sola clase [10]. Un factor crítico en el diseño de un árbol de decisión es su tamaño: debe ser lo suficientemente grande, pero no demasiado; de lo contrario, tiende a aprender los detalles particulares del conjunto de entrenamiento y muestra un pobre rendimiento de generalización. La experiencia ha demostrado que el uso de un valor umbral, para la disminución de impurezas como regla de parada, no siempre conduce a un tamaño óptimo del árbol. Muchas veces, detiene el crecimiento del árbol demasiado pronto o demasiado tarde. El enfoque más utilizado es hacer crecer un árbol hasta un tamaño grande y luego podar sus nodos según un criterio de poda [10]. El tamaño del árbol tiene una importancia significativa para el presente estudio, ya que se trata de un problema de dos clases. Los árboles demasiado grandes o pequeños representarán incorrectamente los vectores de características.

Una vez que se detiene la división, se declara que un nodo es una hoja, y se le asigna una etiqueta de clase 'xj' utilizando la regla de la mayoría. En otras palabras, una hoja t del árbol se asigna a la clase a la que pertenece la mayoría de los vectores 'Xt'. En el caso del árbol de decisión Boosted, el procedimiento normal de aprendizaje recursivo del clasificador estructurado en árbol consiste en dividir el conjunto de fuentes del nodo padre en subconjuntos para los nodos hijos basados en la prueba de clasificación del nodo padre. El problema potencial de un clasificador estructurado en forma de árbol es que el nodo de un clasificador estructurado en forma de árbol pierde la información de distribución de todo el conjunto de datos y es muy susceptible de 'overfitting'. La adición de boosting a un árbol de decisión como medio para mejorar la precisión de la predicción se conoce como boosting adaptativo [10]. El refuerzo adaptativo se basa en un algoritmo de aprendizaje de

un clasificador de árboles de decisión a lo largo de una serie repetida de ensayos: t = 1, ..., T. Un posible enfoque es seleccionar un mejor peso y estructura de árbol a partir de la distribución de pesos sobre el conjunto de entrenamiento.

Se construyen muchos clasificadores a partir de un único conjunto de datos de entrenamiento para el refuerzo. Cada clasificador se construye para formar una estructura SDT o un conjunto de reglas utilizando los datos de entrenamiento. Las nuevas clasificaciones se basan en los votos de muchos clasificadores, mientras que las clases predichas y finales se deciden a partir de los votos. El primer paso de este procedimiento de refuerzo es construir una estructura SDT (Single Decisión Tree) o un conjunto de reglas a partir de los datos de entrenamiento. Este clasificador suele contribuir a los errores de algunos casos de los datos. La primera estructura de árbol de decisión genera la clase incorrecta para algunos casos de los datos de entrenamiento. A continuación, se construye el segundo clasificador prestando mayor atención a la clasificación correcta. En consecuencia, el segundo clasificador será diferente del primer clasificador. El tercer paso de construcción del clasificador está comparativamente más centrado, aunque también cometerá errores en algunos casos. Al establecer el número de pruebas de refuerzo por adelantado, el proceso de refuerzo proceso continúa iterativamente actualizando 'Dt(i)'. El paso final del proceso de refuerzo se detiene cuando el clasificador más reciente es extremadamente preciso o inexacto.

En el caso de Decision tree forest (DTF), los bosques aleatorios (RF) son una de las técnicas de aprendizaje por conjuntos más exitosas que han demostrado ser técnicas muy populares y potentes en el reconocimiento de patrones y el aprendizaje automático para la clasificación de altas dimensiones y problemas asimétricos [10]. Un inconveniente asociado a los clasificadores de árbol es su alta varianza. En la práctica, es habitual que un pequeño cambio en el conjunto de datos de entrenamiento dé lugar a un árbol muy diferente. La razón de esto radica en la naturaleza jerárquica de los clasificadores de árboles. Un error que se produce en un nodo cercano a la raíz del árbol se propaga hasta las hojas. Para que la clasificación en árbol sea más estable, se ha inventado una metodología de bosque de decisión (Random Forrest, RF). Un bosque de decisión es un conjunto de árboles de decisión. Puede verse como un clasificador que contiene varios métodos de clasificación o un método pero varios parámetros de trabajo. Un nuevo vector de entrada es clasificado por cada árbol individual del bosque. Cada árbol produce un determinado resultado de clasificación. El bosque de decisión elige la clasificación que tiene más votos entre todos los árboles del bosque. El ensacado, que significa "'bootstrap aggregation"', es un tipo de aprendizaje conjunto [10], con el fin de mejorar la precisión de un clasificador débil creando un conjunto de clasificadores. En este método, el conjunto de entrenamiento de cada clasificador se genera extrayendo aleatoriamente 'N' ejemplos, con reemplazo, siendo 'N', el tamaño del conjunto de entrenamiento original. El sistema de aprendizaje genera un clasificador a partir de la muestra y agrega todos los clasificadores generados a partir de los distintos ensayos para formar el clasificador final. Para clasificar una instancia, cada clasificador registra un voto para la clase a la que pertenece y la instancia se etiqueta como miembro de la clase con más votos. En caso de que más de una clase reciba conjuntamente el máximo número de votos, el ganador se selecciona al azar. Cada árbol del conjunto se cultiva en una réplica bootstrap de los datos de entrada extraída de forma independiente. Las observaciones no incluidas en esta réplica están "fuera de la bolsa" para este

árbol [10]. El error de predicción del conjunto empaquetado se estima calculando las predicciones de cada árbol sobre sus observaciones fuera de bolsa, promediando estas predicciones sobre todo el conjunto para cada observación y luego comparando la respuesta predicha fuera de bolsa con el valor verdadero en esta observación. El ensamblaje funciona reduciendo la varianza de un aprendiz base insesgado, como un árbol de decisión. Esta técnica tiende a mejorar el poder de predicción del de predicción del conjunto, ya que la selección aleatoria de características correlación entre los árboles del conjunto.

Predicción de datos

La predicción es una táctica utilizada para adumbrar los datos que faltan o datos no disponibles. También es un proceso de dos pasos equivalente a técnica de clasificación de datos, la única diferencia es que en lugar de dar las etiquetas a las clases de datos las etiquetas de las clases se anticipan. En el caso del ejemplo, en la 2.4, en lugar de deducir los valores del atributo clase, la técnica de predicción predice el porcentaje que tiene una muestra de estar asignada en la clase 1 o 2 [7].

Agrupación de datos

La agrupación de datos es un arte de almacenar los datos lógicamente similares compuestos en un grupo [7]. Es una técnica autónoma de minería de datos que puede confundirse con la técnica de clasificación pero existe una gran diferencia entre ellas, ya que en la clasificación el modelo predefinido o las etiquetas de clase se aplican a los datos, aquí las clases también se expresan. La agrupación de datos se basa en la ideología de amplificar las similitudes dentro del grupo y atenuar las similitudes entre los grupos, por ejemplo, la 2.5.

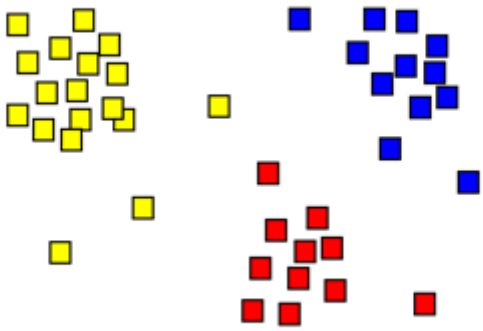


Figure 2.5: Conjunto de datos formando clusters [18]

Valores atípicos

Los objetos de datos que comienzan a desviarse o no están dispuestos a cumplir con el comportamiento genérico mostrado por los otros asociados del modelo de datos en el que se producen se anuncian como atípicos. Un outlier muestra un comportamiento tan diferente que es difícil confiar en que fue engendrado por el mecanismo similar.

La técnica de minería de datos que se ocupa de los valores atípicos se denomina análisis de valores atípicos. En el mundo actual se da más importancia al análisis de valores atípicos, ya que éstos pueden ser indicadores de acontecimientos interesantes que nunca antes se habían conocido, por lo que se presta más atención al análisis de valores atípicos que a la técnica complementaria de minería de datos. Los valores atípicos pueden dividirse en dos grupos: los buenos, que proporcionan información útil que puede conducir a la obtención de nuevos conocimientos; por ejemplo, en 1880, cuando el físico inglés Rayleigh descubrió el gas argón, se le considera un valor atípico, ya que descubrió que el gas nitrógeno de la atmósfera tiene una densidad mayor que el gas nitrógeno preparado en el laboratorio; y el otro grupo es el de los valores atípicos malos, que pueden acompañar a los datos ruidosos o incoherentes [7].

Los valores atípicos son muy difíciles de detectar, ya que implican exploración de enfoques no vistos. La detección de valores atípicos se hace más con el ruido, ya que el ruido distorsiona los objetos normales y, por lo tanto, la distinción entre los objetos de datos normales y los valores atípicos. En general, el análisis de valores atípicos es un proceso de cinco pasos:

1. El conjunto de datos proporciona los datos de entrada sobre los que se realiza la limpieza de datos.

2. Se utilizan varios algoritmos para detectar el valor atípico.

3. El valor atípico se representa de forma adecuada.

4. Describir el perfil del valor atípico detectado.

5. Se exploran las medidas interesantes para la evolución del valor atípico.

Caracterización - cum -discriminación

La caracterización es el proceso de describir la clase en una forma resumida. Las características de la clase, utilizadas para su descripción se denominan descripción del concepto de clase. La descripción de una clase puede realizarse de dos maneras: mediante la caracterización de los datos caracterización que resume la clase y otra, por datos que se utiliza para analizar las diferentes clases, también puede denominarse contraste de clases.

Esta técnica es equivalente a la clasificación [7], sin embargo, la clasificación se concentra en la generación de etiquetas de clase plantadas en un modelo, mientras que la caracterización y la discriminación de los datos se centran en la especialidad de las etiquetas de clase en ausencia de datos de entrenamiento.

2.2 Sensores

Los sensores son dispositivos electrónicos u optoelectrónicos que cumplen la función de detectar un parámetro físico. Existen muchos tipos de sensores, incluidos los que detectan una presencia física como llamas, metales, fugas, niveles de gases y productos químicos, entre otros. Algunos están diseñados para detectar propiedades físicas como la temperatura, la presión o la radiación, mientras que otros pueden detectar el movimiento o la proximidad. Funcionan de diversas maneras según la

aplicación y pueden incluir campos electromagnéticos u ópticos, entre otros. Muchas aplicaciones de una amplia gama de industrias utilizan sensores de muchos tipos para probar, medir y controlar diversos procesos y funciones de las máquinas. Con la llegada del Internet de las cosas (IoT), la necesidad de sensores como herramienta principal para proporcionar una mayor automatización va en incremento.

Los sensores inteligentes son una técnica desarrollada en los años 70, cuando la capacidad de procesamiento, basada en la lectura integrada con el procesamiento de la señal, estaba todavía lejos de la complejidad necesaria en los sistemas avanzados de vigilancia y alerta por rayos infrarrojos (IR), debido a la enorme cantidad de ruido/señales no deseadas emitidas por el escenario operativo, especialmente en las aplicaciones militares. La tecnología de los Sensores Inteligentes se mantuvo restringida dentro de un entorno militar cercano explotando en aplicaciones y prestaciones en los años 90 gracias a las impresionantes mejoras en la lectura y procesamiento de señales integradas logradas por las tecnologías de carga acoplada (CCD). De hecho, los rápidos avances de la tecnología de procesadores de "integración a muy gran escala" (VLSI) y la tecnología de conjuntos de detectores electro-ópticos (EO) en mosaico permitieron desarrollar nuevas generaciones de sensores inteligentes con un procesamiento de señales muy mejorado mediante la integración de microordenadores y otros procesadores de señales VLSI.

"En general, el término sensor inteligente se ha referido como acuñado a mediados de los años 80 y el término "Smart" se atribuye a la inteligencia requerida por tales dispositivos gracias a la integración de la unidad de microcontrolador (MCU), el procesador de señal digital (DSP), y las tecnologías de circuitos integrados de aplicación específica (ASIC) desarrollados por trabajar activamente en dispositivos de silicio más inteligentes para los lados de entrada y salida del sistema de control. Más tarde, el significado de este exitoso término se amplió al de sistema micro-electromecánico (MEMS), utilizado para describir una estructura creada con procesos de fabricación de semiconductores para sensores y actuadores" [15].

Los sensores MEMS se utilizan en muchas aplicaciones y pueden encontrarse en sistemas que van desde la automoción, aplicaciones médicas, químicas, industriales y de consumo. Los avances en la tecnología MEMS han permitido detectar y controlar las condiciones físicas y ambientales a bajo coste. Al dotar de más inteligencia a los sensores, podemos construir mejores sistemas combinando microprocesadores integrados y comunicaciones inalámbricas.

Los sistemas microelectromecánicos (MEMS) son una tecnología de proceso utilizada para crear diminutos dispositivos integrados que combinan componentes mecánicos y eléctricos. Se fabrican mediante técnicas convencionales de procesamiento por lotes y su tamaño puede oscilar entre unos pocos micrómetros y milímetros. El tamaño de los dispositivos MEMS ha ido disminuyendo con el tiempo. El escalado de la tecnología ha permitido que los chips MEMS se reduzcan de $1cm^2$ en 1990 a menos de $1mm^2$ en 2014.

La ventaja de la tecnología MEMS es el cambio de paradigma en la miniaturización de las funciones mecánicas y eléctricas en la misma matriz. Hoy en día, los componentes mecánicos MEMS pueden fabricarse con un excelente rendimiento, alta fiabilidad y buena producción. La figura 2.6 muestra una microfotografía de un giroscopio MEMS fabricado en un proceso de silicio por lotes.

Los sensores físicos, que incluyen acelerómetros y giroscopios, constituyen la mayor parte del mercado de sensores MEMS. Los acelerómetros MEMS son sen-

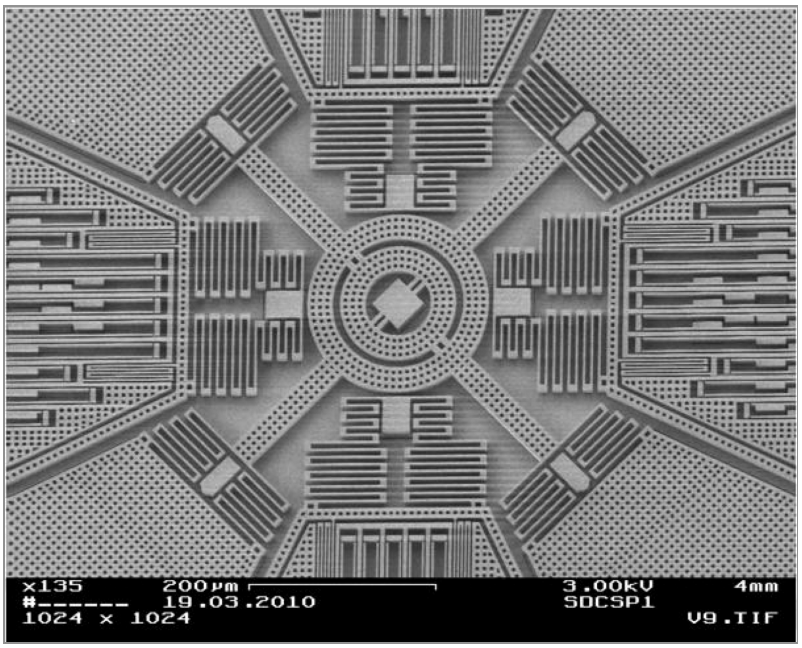


Figure 2.6: Microfotografía de un giroscopio MEMS [31].

sores capaces de detectar la aceleración lineal, mientras que un giroscopio es capaz de medir las velocidades angulares alrededor de uno o más ejes. Un dispositivo inteligente moderno incluye una combinación de acelerómetros y giroscopios, lo que permite seguir y capturar movimientos en un espacio tridimensional. Esto permite a los desarrolladores de sistemas ofrecer experiencias de usuario más envolventes y sistemas de navegación más precisos [31].

Los microsensores y microactuadores son el núcleo de un dispositivo o sistema MEMS. Un microsensor detecta cambios en el entorno del sistema; una parte "inteligente" procesa la información detectada por el sensor y toma una decisión en forma de señal; y un microactuador actúa sobre esta señal para crear algún tipo de cambio en el entorno. Los componentes microelectrónicos constituyen la mayor parte de la parte inteligente del dispositivo.

Los sensores y actuadores se denominan en general transductores, cuya salida es una señal eléctrica. Muchos de los sensores y actuadores MEMS que se han desarrollado dentro de la industria microelectrónica no implican ninguna técnica especial de micromecanizado; se basan en circuitos integrados convencionales que, mediante mecanismos inherentes, detectan la luz, la temperatura, etc. Sin embargo, muchos de ellos pueden mejorarse con el uso de MEMS [15].

El diseño de los sensores inteligentes plantea muchos retos, como el tamaño reducido, la baja disipación de energía, el alto rendimiento y la robustez. La restricción más difícil de cumplir aquí es el bajo consumo de energía. Esto se debe a que, a medida que el tamaño físico del sistema de sensores disminuye, también disminuye el consumo energético. La mayor parte del consumo de energía se destina al sensor seguido del enlace de comunicación inalámbrica. El sistema también incluye un microprocesador que procesa y almacena los datos del sensor. Para ahorrar energía, el enlace inalámbrico suele para enviar paquetes a intervalos regulares y luego se pone en modo de espera.

Existe una enorme variedad de sensores mecánicos directos que han sido o podrían ser micromecanizados en función de su mecanismo de detección (normal-

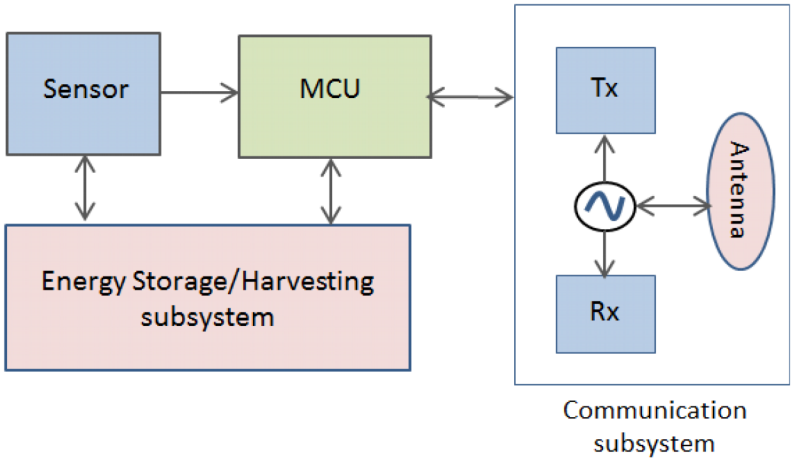


Figure 2.7: Esquema de un dispositivo con varios sensores [31].

mente piezoresistivo, piezoelecto o capacitivo) y de los parámetros detectados (normalmente deformación, fuerza y desplazamiento). Algunos tipos de sensores más comunes que se suelen emplear en la automatización de los procesos se exponen a continuación[16].

2.2.1 Sensores piezorresistivos

Como resultado del efecto piezorresistivo (definido como el cambio en la resistividad del material con la tensión aplicada), los cambios en la dimensión del calibre dan lugar a cambios proporcionales en la resistencia del sensor. El efecto piezorresistivo en los semiconductores es considerablemente mayor que en los metales tradicionales, lo que convierte al silicio en un excelente sensor de deformación. Los piezorresistores MEMS se fabrican fácilmente utilizando silicio a gran escala dopado con impurezas de tipo p o de tipo n[30].

2.2.2 Sensores piezoeléctricos

Los sensores piezoeléctricos utilizan el efecto piezoeléctrico en el que una tensión (o fuerza) aplicada sobre un cristal piezoeléctrico da lugar a una diferencia de potencial a través del cristal. Del mismo modo, si el cristal se somete a una diferencia de potencial, se produce un desplazamiento o una deformación. Este efecto puede utilizarse para detectar la tensión mecánica (es decir, el desplazamiento) y como mecanismo de accionamiento, aunque los desplazamientos son pequeños incluso para tensiones elevadas. Los materiales piezoeléctricos más comunes utilizados para aplicaciones MEMS son el cuarzo, el titanato de circonato de plomo (PZT), el fluoruro de polivinilideno (PVDF) y ZnO, siendo el PVDF y el ZnO los más comunes. El silicio no es piezoeléctrico; Por lo tanto, hay que depositar una fina película de un material adecuado en los dispositivos[30].

2.2.3 Sensores capacitivos

La detección capacitiva (o electrostática) es uno de los mecanismos de detección de precisión más importantes (y ampliamente utilizados) e incluye una o más placas

conductoras fijas con una o más placas conductoras móviles. La detección capacitiva se basa en la ecuación básica del condensador de placas paralelas. Como la capacidad es inversamente proporcional a la distancia entre las placas, la detección de desplazamientos muy pequeños es extremadamente precisa[30].

- Los acelerómetros detectan la aceleración utilizando una masa de prueba suspendida sobre la que puede actuar una aceleración externa. En caso de aceleración (o deceleración), se genera una fuerza (F=ma) sobre la masa de prueba que provoca un desplazamiento. La fuerza o el desplazamiento suelen medirse por métodos piezorresistivos y capacitivos [30].

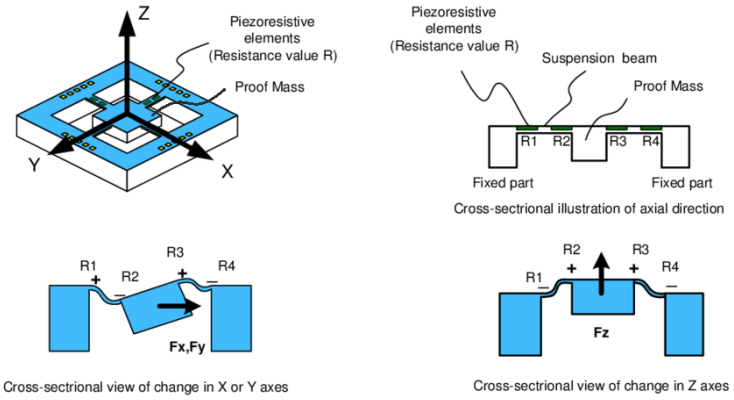


Figure 2.8: Masa de prueba suspendida en un acelerómetro piezoresistivo [1].

- Un giroscopio es un dispositivo que mide la velocidad de rotación y detecta el movimiento angular inercial. Por ello, puede encontrarse, por ejemplo, en aplicaciones de transporte, navegación y aplicaciones de guiado de misiles. Se basa en la medición de la influencia de la fuerza de Coriolis sobre un cuerpo en un marco de rotación. Los giroscopios MEMS suelen utilizar estructuras vibratorias debido a la dificultad de micromecanizar piezas giratorias con suficiente masa útil.

- Los sensores de presión MEMS suelen basarse en membranas finas con con cavidades selladas llenas de gas o de vacío en un lado de la membrana y la presión a medir en el otro lado. medir en el otro lado. Las técnicas de medición de la deflexión de la membrana piezoresistiva y capacitiva son las más utilizadas en los sensores de presión comerciales.

- El termopar es probablemente el transductor de temperatura más común. Consiste en una unión entre dos materiales diferentes y mide la tensión dependiente de la temperatura que surge a través de la unión. Los materiales semiconductores suelen presentar un mejor efecto termoeléctrico que los metales. Los termopares se han utilizado en una gran variedad de sensores MEMS en una disposición de matriz denominada termopila[30].

2.3 Message Queuing Telemetry Transport (MQTT)

MQTT fue definido según sus autores como un protocolo ligero orientado a eventos y mensajes que permite a los dispositivos comunicarse de forma asíncrona y eficiente a través de redes limitadas con sistemas remotos. Este protocolo, heredero del modelo de comunicación del intercambio de mensajes, está basado en un broker que cumple la función de intermediario entre los productores de los mensajes y los consumidores de estos [26].

A la hora de diseñar este protocolo, los autores se centraron en la posibilidad de conectar de manera fácil el mundo físico al mundo cibernético (conexión M2M). Además, puede soportar redes poco fiables caracterizadas por un ancho de banda reducido y/o una latencia elevada. El acoplamiento ligero que proporciona este protocolo es capaz de soportar entornos dinámicos donde grandes cantidades de mensajes han de intercambiarse de formas no previstas de manera anticipada. Este protocolo también permite la definición de niveles de Quality of Service en los mensajes, de tal manera que se pueda afinar el compromiso entre ancho de banda, disponibilidad y garantía de entrega. Capaz de soportar grandes cantidades de dispositivos (10000 clientes MQTT), fue diseñado con un enfoque simple para los desarrolladores y se caracteriza por ser de código abierto [26].

Hay tres partes en la arquitectura MQTT:

- Broker MQTT - Todos los mensajes pasados del cliente al servidor deben ser enviados a través del broker.
- Servidor MQTT - La API actúa como un servidor MQTT. El servidor MQTT se encargará de publicar los datos a los clientes.
- Cliente MQTT - Cualquier cliente de terceros que desee suscribirse a los datos publicados por la API, se considera un cliente MQTT.

Los clientes MQTT necesitan conectarse al Broker para publicar mensajes o suscribirse a asuntos. Supongamos que tenemos conectado un dispositivo que envía

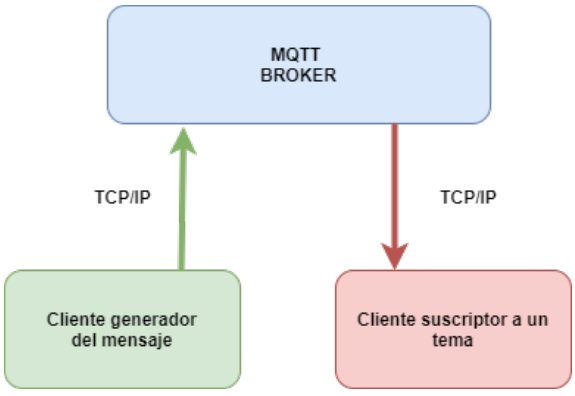


Figure 2.9: MQTT communication structure.

datos de distintos sensores mediante el protocolo MQTT. El broker recoge los datos del dispositivo, clasifica los asuntos de los mensajes en función de los sensores que envían datos y reenvía los mensajes a cualquier otro cliente que se suscriba a un asunto disponible.

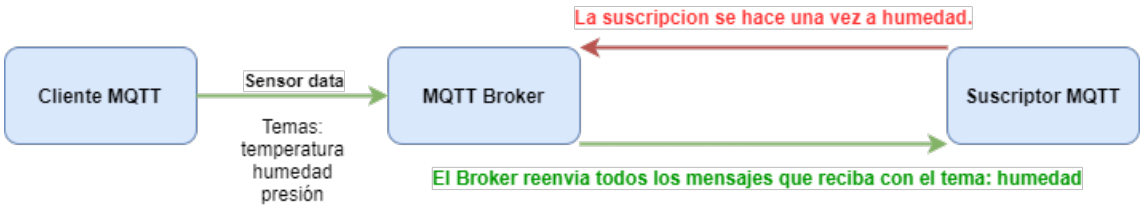


Figure 2.10: Flujo de datos desde el módulo de la API a los clientes externos.

2.3.1 Broker MQTT

El Broker MQTT es un servicio encargado de recibir todos los mensajes de los clientes, filtrar los mensajes, decidir que clientes están interesados en determinados asuntos y reenviar los mensajes a todos los clientes suscritos al mismo asunto.

Todos los mensajes publicados pasan por el broker. El broker genera el ID de cliente y el ID de mensaje, mantiene la cola de mensajes y publica el mensaje.

2.3.2 Topic MQTT

Un asunto (topic) es una cadena de caracteres (UTF-8). Utilizando esta cadena, el Broker filtra los mensajes para todos los clientes conectados. Un asunto puede consistir en uno o más niveles ordenados de manera jerárquica. La barra oblicua (separador de nivel de asunto) se utiliza para separar cada nivel de asunto.

Figure 2.11: Ejemplo visual de temas en MQTT.

Los asuntos están enmarcados de tal manera que proporcionan opciones para que el usuario se suscriba a cualquier nivel de tema o a los datos individuales del nivel del sensor. Al suscribirse a cada nivel de datos de los sensores, el cliente debe especificar la jerarquía de los ID. Por ejemplo, para suscribirse a los datos del sensor de nivel 3, el cliente debe especificar el ID del nivel 1, el ID del nivel 2, y el ID de nivel 3. El usuario puede suscribirse a cualquier tipo de sensor especificando el rol del sensor como última parte del tema. Si el usuario no especifica el rol, el cliente se suscribirá a todos los tipos de sensores en ese nivel particular. El usuario también puede especificar el ID del sensor al que desea suscribirse. En ese caso, necesitan especificar toda la jerarquía del sensor, empezando por el id del proyecto y terminando con el id del sensor.

Funciones soportadas por MQTT

- Autenticación: MQTT proporciona la autenticación de cada usuario que pretenda publicar o suscribirse a unos datos determinados. El nombre de usuario

y la contraseña se almacenan en la base de datos de la API, en una colección separada llamada 'mqtt'. Mientras nos conectamos al broker MQTT, proporcionamos el nombre de usuario y la contraseña, y el broker MQTT validará las credenciales basándose en los valores presentes en la base de datos.

- Calidad de servicio: El nivel de calidad de servicio (QoS) es el nivel de garantía de transmisión que se requiere del Broker en los mensajes. Hay 3 niveles de QoS en MQTT:

  1. QOS = 0 El mensaje se entrega como máximo una vez, o no se entrega.
  2. QOS = 1 El mensaje siempre se entrega al menos una vez.
  3. QOS = 2 El mensaje siempre se entrega exactamente una vez.

- Mensaje retenido: MQTT también tiene una función de retención de mensajes. Cuando un cliente se suscribe a un tema, el broker hace coincidir el tema con un mensaje retenido. Los clientes recibirán los mensajes inmediatamente si el tema y el mensaje retenido coinciden. Los brokers sólo almacenan un mensaje retenido para cada tema.

- Mensajes duplicados: Si el publicador de un mensaje no recibe un aviso del paquete publicado, seguirá enviando el mismo mensaje que tendrá el ID y la información original.

Chapter 3

Desarrollo Sensor

3.1 Sistema

Antes de comenzar a explicar las partes clave que componen el sistema, me gustaría mencionar los módulos y los esquemas de flujo de los datos que representan el funcionamiento general.El flujo de los datos que se genera en el dispositivo en sí debido a los sensores de los que dispone se puede direccionar hacia dos caminos diferentes.

La primera opción es que los datos una vez generados se envíen a un dispositivo móvil mediante una conexión Bluetooth. Los datos una vez obtenidos en el teléfono mediante el uso de la aplicación disponible para este dispositivo se pueden reenviar a una base de datos utilizando el protocolo de mensajería MQTT. Para poder reenviar los datos a una base de datos en la red, hace falta utilizar la aplicación que este dispositivo ofrece para el móvil. El firmware del dispositivo funciona en en paralelo con la aplicación móvil. Es decir, una vez que se generan los datos el firmware y la aplicación móvil comprueban el estado de la conexión Bluetooth. Si esta conexión está activa los datos se envían, en caso contrario los datos se eliminan del dispositivo.

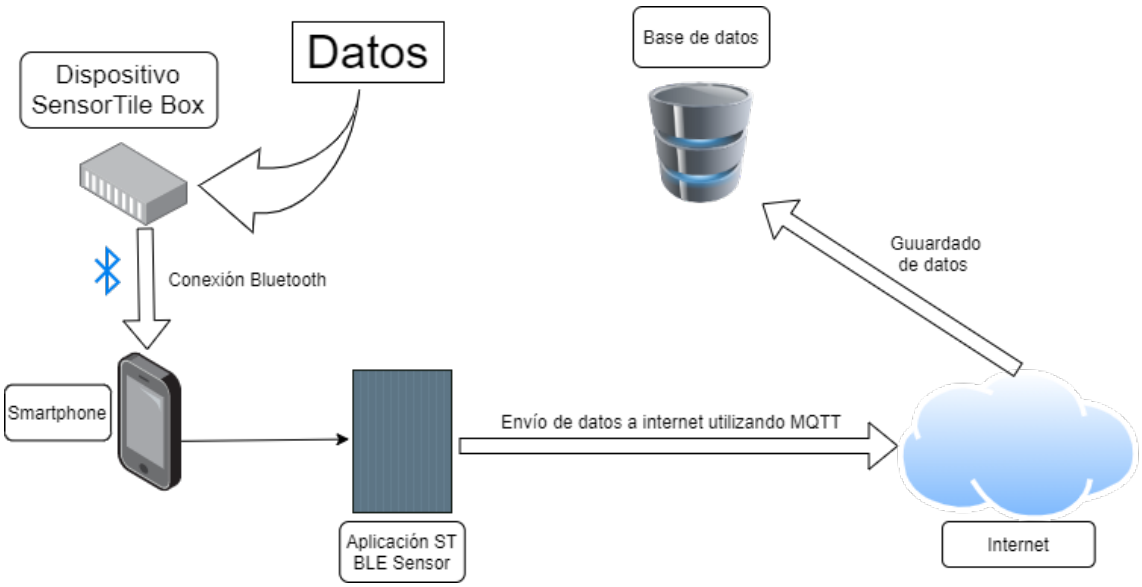


Figure 3.1: Esquema de flujo de datos y guardado en la base de datos de la red.

La segunda opción es guardar los datos generados por el dispositivo en la memoria

interna que este dispone. Esta opción, aunque parece más sensata a la hora del uso, tiene un inconveniente. Necesita tener el teléfono móvil siempre conectado al dispositivo a pesar de que los datos no se envíen al smartphone. La conexión con el teléfono sirve para arrancar el proceso de guardado de los datos, parar ese proceso y monitorizar la conexión Bluetooth entre ambos dispositivos.

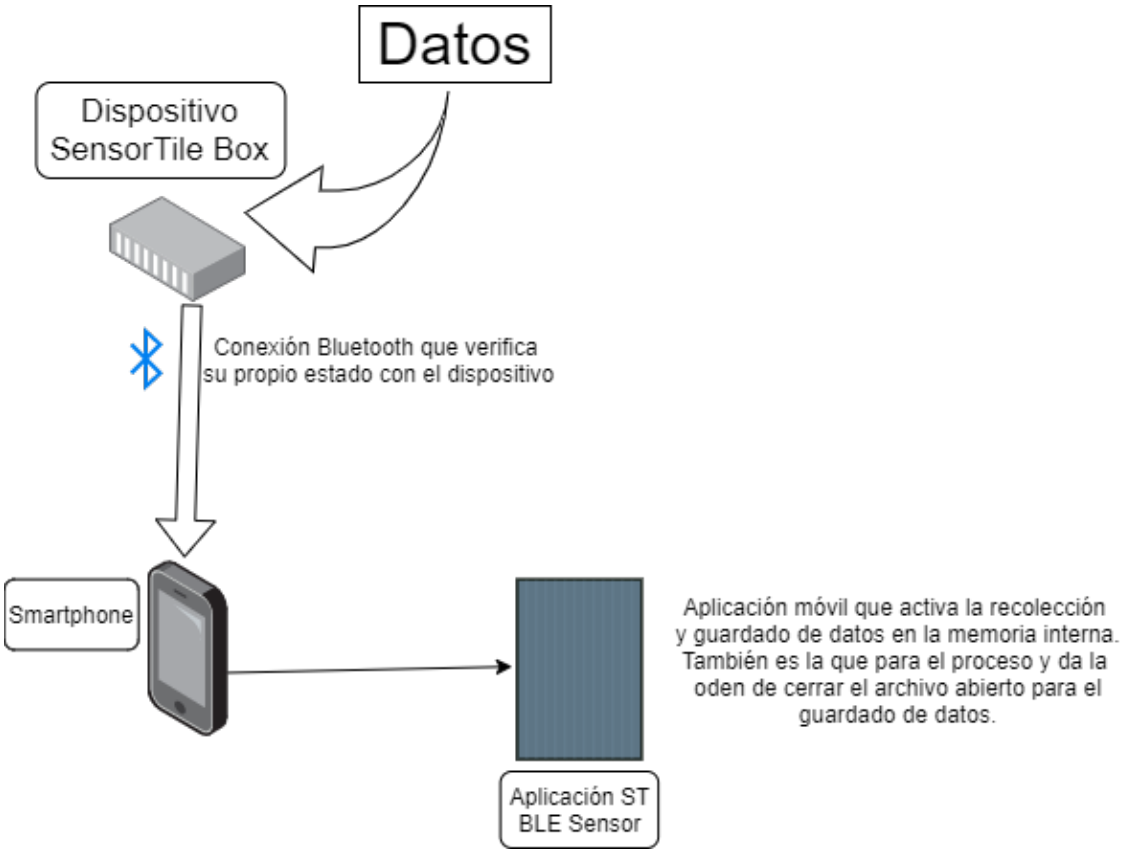


Figure 3.2: Esquema de flujo de datos y guardado en el propio dispositivo

Como se puede apreciar en la imagen 3.3 SensorTile Box dispone de un módulo para conexiones Bluetooth. Este módulo permite hacer conexiones con el dispositivo móvil utilizando la aplicación que ofrece el fabricante. El módulo Bluetooth es utilizado para enviar comandos al dispositivo, parar o comenzar procesos de registro y guardado de datos. Este módulo es controlado por el firmware del dispositivo que está cargado en el microcontrolador. El microcontrolador se encarga de gestionar todos los periféricos de los que dispone. Entre estos se incluyen los sensores, la tarjeta microSD y otro microcontrolador diseñado para dotar al dispositivo de inteligencia artificial.

El microcontrolador encargado de la inteligencia artificial se denomina STM32Cube-AI. Este periférico tiene la capacidad de conversión automática de la red neuronal pre-entrenada y la integración de la biblioteca optimizada generada en el proyecto del usuario. En otras palabras, este dispositivo se encarga de procesar un fichero de configuración generado por un árbol de decisión que a su vez se genera a partir de los datos capturados del dispositivo con el propósito del aprendizaje automático.

Cabe destacar que para utilizar el microcontrolador STM32Cube-AI es necesario preparar los datos capturados desde un principio. Se debe capturar una cantidad suficiente de datos representativos sobre el fenómeno que se está modelando. Al-

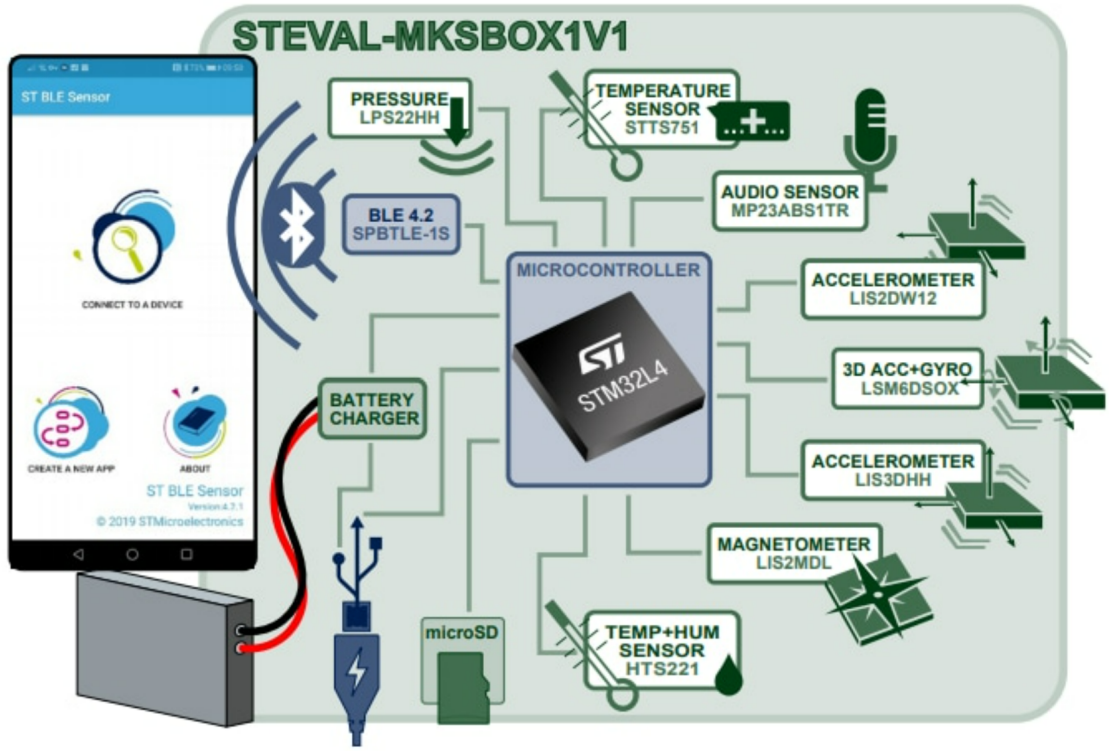


Figure 3.3: Esquema general del duspositivo [2].

gunos ejemplos de parámetros físicos son la aceleración, la temperatura, el sonido y la presión atmosférica, dependiendo de la aplicación. Los datos capturados con el dispositivo necesitan además estar etiquetados para el llamado "aprendizaje supervisado". Los conjuntos de datos deben ser caracterizados para que las diferentes salidas puedan ser clasificadas correctamente. Este conjunto clasificado es la "verdad básica" que se utilizará para entrenar la red neuronal artificial (RNA) y luego validarla. El desarrollador debe decidir el tipo de topología que debe tener la RNA para poder aprender mejor de los datos y proporcionar una salida útil para la aplicación objetivo. Por lo general, los desarrolladores emplean marcos de aprendizaje profundo conocidos para diseñar y entrenar topologías de redes neuronales artificiales.

El entrenamiento de la RNA consiste en hacer pasar los conjuntos de datos por la red neuronal de forma iterativa para que las salidas de la red puedan minimizar los criterios de error deseables. La definición, el entrenamiento y la comprobación de la RNA se realizan normalmente utilizando marcos de aprendizaje profundo estándar. Esto se suele hacer en una plataforma informática potente, con memoria y potencia de cálculo prácticamente ilimitadas, para permitir muchas iteraciones en un corto periodo de tiempo. El resultado de este entrenamiento es la red neuronal artificial preentrenada.

El siguiente paso es integrar la RNA preentrenada en el microcontrolador (MCU) mediante una optimización que reduzca los requisitos de cómputo y memoria (y por ende la complejidad).Esta parte es muy fácil e intuitiva gracias a la herramienta de software STM32Cube.AI, que permite la conversión rápida y automática de las RNA preentrenadas en código optimizado que puede ejecutarse en una MCU. La herramienta guía a los usuarios a través de la selección de la MCU adecuada y

proporciona información rápida sobre el rendimiento de la red neuronal en la MCU elegida.

Por último, despliegue la RNA integrada en una MCU en su aplicación. En este caso, ST también facilita a los diseñadores la creación rápida de prototipos de sus aplicaciones innovadoras gracias a los paquetes de software integrados: los paquetes de funciones. Estos paquetes son ejemplos integrales que incorporan una combinación de controladores de bajo nivel, bibliotecas de middleware y aplicaciones de muestra reunidas en un único paquete de software. Los desarrolladores pueden partir fácilmente de estos ejemplos y realizar modificaciones para adaptarlos a su aplicación específica. Dos ejemplos habilitados para la IA son los paquetes de funciones de audio y de captura y procesamiento de movimiento.

3.2 Descripción del sistema

El dispositivo STEVAL-MKSBOX1V1 (SensorTile.box) es un kit de caja que se ha empleado para la recopilación de datos y desarrollo de este trabajo. Es un dispositivo listo para utilizar con plataforma de sensores inalámbricos, IoT y tecnología wearable que permite la recolección de datos de los sensores que dispone y desarrollo de aplicaciones. La placa SensorTile.box cabe en una pequeña caja de plástico con una batería recargable y dispone de una aplicación para smartphones (ST BLE Sensor) que puede ser utilizada para conectarse a la placa vía Bluetooth.

La aplicación permite comenzar inmediatamente la recolección de datos, así como probar algunas demostraciones de funcionalidad que ofrece el dispositivo. Cabe destacar que la aplicación para smartphones muestra las funcionalidades de la placa solo si estas fueron propiamente habilitadas de antemano en el código del firmware. Por lo tanto, la funcionalidad de la aplicación depende de manera directa de la funcionalidad que fue programada de manera previa en la placa. El Modo Experto de la aplicación ofrece la posibilidad de construir aplicaciones personalizadas a partir de una selección de sensores SensorTile.box, parámetros de funcionamiento, tipos de datos y salidas, y funciones especiales y algoritmos disponibles.

SensorTile.box incluye una interfaz de programación y depuración de firmware que permite a los desarrolladores dedicarse a un desarrollo de código de firmware más complejo utilizando el Entorno de Desarrollo Abierto STM32 (STM32 ODE), que incluye un paquete de funciones de IA de detección con bibliotecas de redes neuronales.

Figure 3.4: SensorTile.Box [34]

SensorTile.box consiste de un microcontrolador que tiene acceso a distintos periféricos y sensores que va gestionando a la vez en función de cómo es programado. Un microcontrolador (a veces llamado MCU o unidad de microcontrolador) es un circuito integrado (IC) que suele utilizarse para una aplicación específica y está diseñado para realizar determinadas tareas. Los productos y dispositivos que deben controlarse automáticamente en determinadas situaciones, como los electrodomésticos, las herramientas eléctricas, los sistemas de control de motores de automóviles y los ordenadores, son grandes ejemplos, pero los microcontroladores van mucho más allá de estas aplicaciones.

Esencialmente, un microcontrolador recoge información de entrada, procesa esta información y emite una determinada acción basada en la información recogida. Los microcontroladores suelen funcionar a velocidades más bajas, en torno al rango de 1MHz a 200 MHz, y tienen que estar diseñados para consumir menos energía porque están incrustados dentro de otros dispositivos que pueden tener mayores consumos de energía en otras áreas [23].

Un microcontrolador puede ser visto como un pequeño ordenador, y esto es debido a los componentes esenciales dentro de él; la Unidad Central de Procesamiento (CPU), la Memoria de Acceso Aleatorio (RAM), la Memoria Flash, la Interfaz de Bus Serial, los Puertos de Entrada/Salida (Puertos I/O), y en muchos casos, la Memoria Eléctrica Programable de Sólo Lectura (EEPROM).

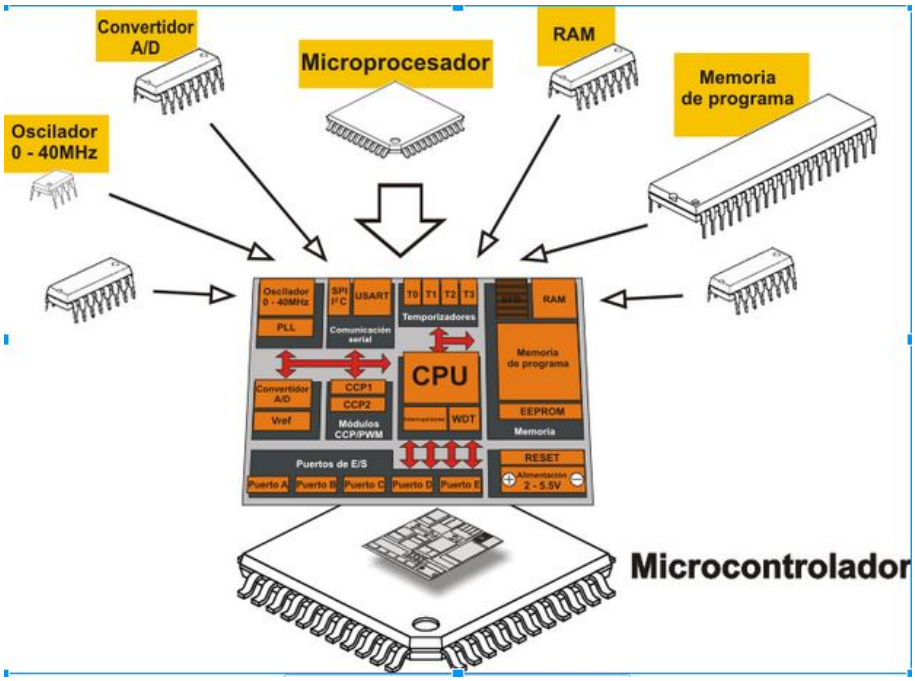


Figure 3.5: Estructura de un microcontrolador [23]

Un microcontrolador consta de una unidad central de procesamiento (CPU), memoria no volátil, memoria volátil, periféricos y circuitos de apoyo. La CPU realiza operaciones aritméticas, gestiona el flujo de datos y genera señales de control de acuerdo con la secuencia de instrucciones creada por el programador. La complejísima circuitería necesaria para la funcionalidad de la CPU no es visible para el diseñador. De hecho, gracias a los entornos de desarrollo integrados y a lenguajes de alto nivel como el C, escribir código para microcontroladores suele ser una tarea relativamente sencilla.

La memoria no volátil se utiliza para almacenar el programa del microcontrolador, es decir, la lista (a menudo muy larga) de instrucciones en lenguaje de máquina que indican a la CPU exactamente lo que debe hacer. En lugar de "memoria no volátil", se suele utilizar la palabra "Flash" (que se refiere a una forma específica de almacenamiento de datos no volátil).

La memoria volátil (es decir, la RAM) se utiliza para el almacenamiento temporal de datos. Estos datos se pierden cuando el microcontrolador pierde energía. Los registros internos también proporcionan almacenamiento temporal de datos, pero no pensamos en ellos como un bloque funcional separado porque están integrados en la CPU.

Utilizamos la palabra "periférico" para describir los módulos de hardware que ayudan a un microcontrolador a interactuar con el sistema externo. A continuación se exponen las distintas categorías de periféricos y algunos ejemplos.

- Convertidores de datos: convertidor analógico-digital, convertidor digital-analógico, generador de tensión de referencia
- Generación de reloj: oscilador interno, circuito de accionamiento de cristal, bucle de bloqueo de fase
- Temporización: temporizador de propósito general, reloj en tiempo real, contador de eventos externos, modulación por ancho de pulso
- Procesamiento de señales analógicas: amplificador operacional, comparador analógico
- Entrada/salida: circuito de entrada y salida digital de propósito general, interfaz de memoria paralela
- Comunicación en serie: UART, SPI, I2C, USB

Los microcontroladores incorporan una serie de bloques funcionales que no pueden clasificarse como periféricos porque su propósito principal no es controlar, supervisar o comunicarse con componentes externos. Sin embargo, son muy importantes, ya que ayudan al funcionamiento interno del dispositivo, simplifican la implementación y mejoran el proceso de desarrollo. Los circuitos de depuración permiten al diseñador supervisar cuidadosamente el microcontrolador mientras ejecuta las instrucciones. Se trata de un método importante, y a veces indispensable, para detectar errores y optimizar el rendimiento del firmware. Las interrupciones son un aspecto muy valioso de la funcionalidad del microcontrolador. Las interrupciones son generadas por eventos externos o internos basados en hardware, y hacen que el procesador responda inmediatamente a estos eventos ejecutando un grupo específico de instrucciones. Un módulo de generación de reloj puede considerarse un periférico si está destinado a producir señales que se utilizarán fuera del chip, pero en muchos casos el propósito principal del oscilador interno de un microcontrolador es proporcionar una señal de reloj para la CPU y los periféricos. Los osciladores internos suelen tener una baja precisión, pero en aplicaciones que pueden tolerar esta baja precisión, son una forma conveniente y efectiva de simplificar el diseño y ahorrar espacio en la placa.

Los microcontroladores pueden incorporar varios tipos de circuitos de alimentación. Los reguladores de tensión integrados permiten generar en el chip las tensiones de

alimentación necesarias, los módulos de gestión de la alimentación pueden utilizarse para reducir considerablemente el consumo de corriente del dispositivo durante los estados inactivos, y los módulos de supervisión pueden colocar el procesador en un estado de reinicio estable cuando la tensión de alimentación no es lo suficientemente alta para garantizar un funcionamiento fiable.

Sensor de temperatura (STTS751)

El dispositivo compacto dispone de un sensor principal digital de temperatura. El sensor de temperatura digital se comunica a través de un bus de 2 hilos. La temperatura se mide con una resolución configurable por el usuario entre 9 y 12 bits. A 9 bits, el tamaño de paso más pequeño es de 0,5 °C, y a 12 bits, de 0,0625 °C. Con la resolución por defecto (10 bits, 0,25 °C/LSB), el tiempo de conversión es nominalmente de 21 milisegundos. Dispone de una salida que se utiliza para indicar una condición de alarma que indica que la medición se encuentra por fuera de los límites máximos y mínimos establecidos por el usuario en el programa. Cuando dicha salida se activa, el procesador puede comunicarse con el sensor solicitar la dirección del origen de la señal. STTS751 es un dispositivo de 6 pines que soporta direcciones esclavas configurables por el usuario. A través de la resistencia de pull-up en el pin Addr/Therm, se puede especificar una de las cuatro direcciones esclavas diferentes. Dos números de pedido (STTS751-0 y STTS751-1) proporcionan dos conjuntos diferentes de direcciones esclavas, con lo que el total disponible es de ocho. De este modo, hasta ocho dispositivos pueden compartir el mismo bus de dos hilos sin ambigüedad, permitiendo así la monitorización de múltiples zonas de temperatura en una aplicación. La interfaz de dos hilos puede soportar velocidades de transferencia de hasta 400 kHz [4].

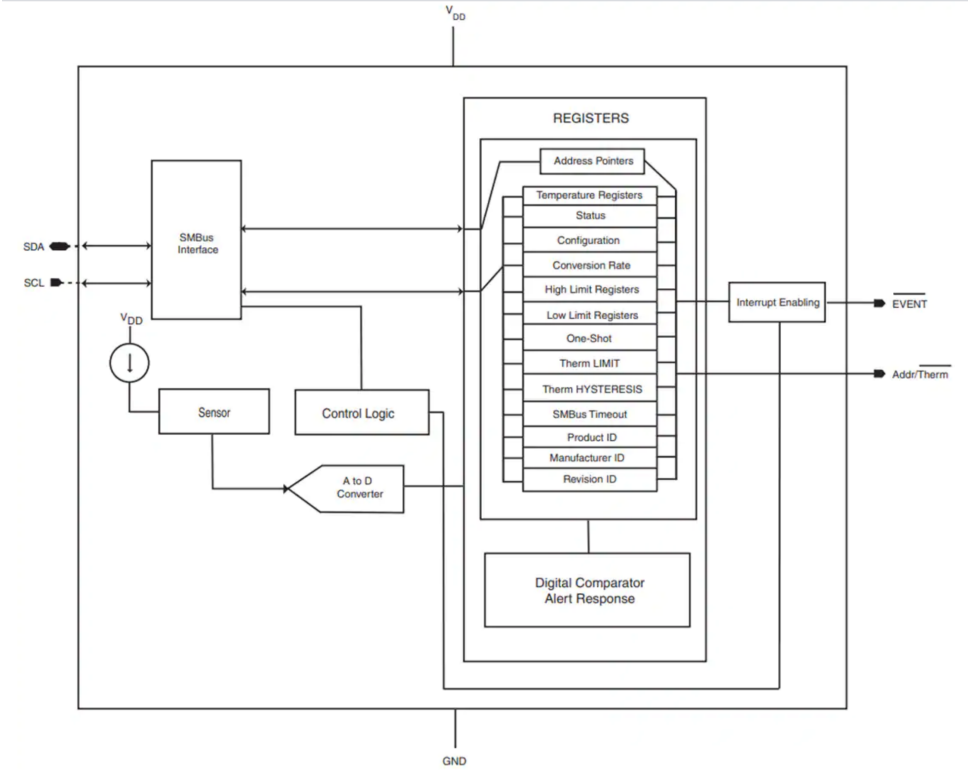


Figure 3.6: Diagrama de bloques del sensor de temperatura (STTS751) [28]

Sensor inercial INEMO (LSM6DSOX)

El sensor LSM6DSOX es un conjunto de un acelerómetro digital 3D y un giroscopio digital 3D con dos modos: alto rendimiento, que ofrece mayores prestaciones a cambio de un mayor requerimiento energético de 0,55 mA de intensidad, y bajo consumo, con unas funciones mínimas siempre activas para una experiencia de movimiento óptima para el consumidor. El periférico de detección es compatible con los principales requisitos del sistema operativo y ofrece sensores reales, virtuales y por lotes con 9 kbytes para la agrupación dinámica de datos [22]. El LSM6DSOX tiene un rango de aceleración a escala completa de ±2/±4/±8/±16 g y un rango de velocidad angular de ±125/±250/±500/±1000/±2000 dps [6].

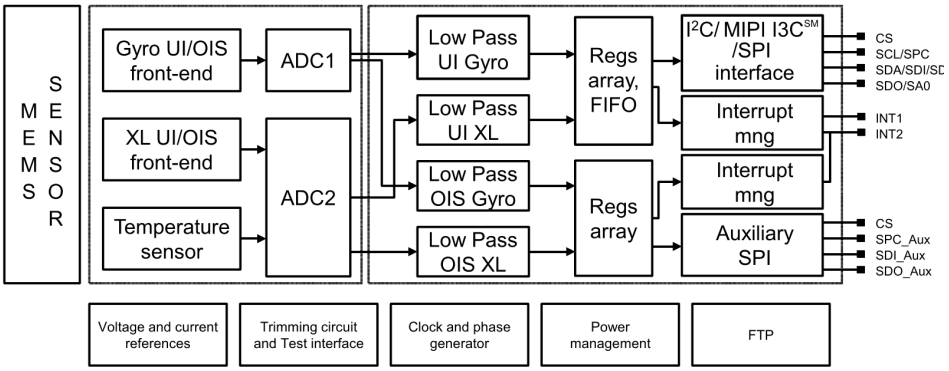


Figure 3.7: Diagrama de bloques del sensor inercial INEMO. Muestra los modos generales de funcionamiento y configuración. [28]

En el LSM6DSOX, el acelerómetro puede configurarse en cinco modos de funcionamiento diferentes: apagado, ultrabajo consumo, bajo consumo, modo normal y modo de alto rendimiento. El modo de funcionamiento seleccionado depende del valor del bit XL_HM_MODE en $CTRL6\_C$. Si XL_HM_MODE está ajustado a '0', el modo de alto rendimiento es válido para todas las frecuencias desde 12,5 Hz hasta 6,66 kHz. Para habilitar el modo de bajo consumo y el modo normal, el bit XL_HM_MODE debe estar a '1'. El modo de baja potencia está disponible para las frecuencias más bajas (1,6, 12,5, 26, 52 Hz), mientras que el modo normal está disponible para las frecuencias iguales a 104 y 208 Hz.

El giroscopio de este dispositivo MEMS puede configurarse en cuatro modos de funcionamiento diferentes: apagado, bajo consumo, modo normal y modo de alto rendimiento. El modo de funcionamiento seleccionado depende del valor del bit G_HM_MODE en $CTRL7\_G$. Si G_HM_MODE está ajustado a '0', el modo de alto rendimiento es válido para todas las frecuencias desde 12,5 Hz hasta 6,66 kHz. Para habilitar el modo de bajo consumo y el modo normal, el bit G_HM_MODE debe ponerse a '1'. El modo de bajo consumo está disponible para las frecuencias más bajas (12,5, 26, 52 Hz), mientras que el modo normal está disponible para las frecuencias iguales a 104 y 208 Hz.

Acelerómetro

LIS2DW12 es un acelerómetro lineal de tres ejes de alto rendimiento y muy bajo consumo, perteneciente a la familia "femto". Tiene escalas completas seleccionables

por el usuario de ±2g/±4g/±8g/±16g y es capaz de medir aceleraciones con velocidades de salida de datos de 1,6 Hz a 1600 Hz. LIS2DW12 tiene un búfer integrado de 32 niveles de primera entrada, primera salida (FIFO) que permite al usuario almacenar datos para limitar la intervención del procesador anfitrión. Además, dispone de otro acelerómetro (LIS3DHH) lineal de tres ejes de muy alta resolución y bajo ruido. Este sensor tiene una escala completa de ±2,5 g y es capaz de proporcionar las aceleraciones medidas a la aplicación a través de una interfaz digital [5].

La LIS2DW12 cuenta con un motor interno dedicado a procesar la detección de movimiento y aceleración, incluyendo la caída libre y los movimientos bruscos. También incluye la detección de movimiento fijo, la detección de formato de retrato/paisaje y la alineación 6D/4D. La LIS2DW12 está disponible en una pequeña y estrecha carcasa Land Grid Array (LGA) de plástico y su funcionamiento está garantizado en un amplio rango de temperaturas de -40°C a +85°C [5].

Magnetómetro

Uno de los sensores del dispositivo SensorTile.Box es un magnetómetro (LIS2MDL). Este sensor magnético digital de 3 ejes es de muy bajo consumo y alto rendimiento. Este magnetómetro LIS2MDL tiene un rango dinámico de campo magnético de ± 50 Gauss, tres canales de campo magnético, interfaces en serie SPI/I^2 C, un generador de interrupciones programable y una salida de datos de 16 bits. La interfaz de bus serie I 2 C admite un modo estándar (100 kHz), modo rápido (400 kHz), modo rápido plus (1 MHz) y alta velocidad (3,4 MHz). [25].

Barómetro

El sensor LPS22HH del dispositivo, es un dispositivo que funciona como barómetro de salida digital. LPS22HH MEMS es un sensor de presión absoluta piezorresistivo ultracompacto que actúa como un barómetro de salida digital. El elemento sensor detecta la presión absoluta y consiste en una membrana suspendida que fue fabricada mediante un proceso desarrollado específicamente para este propósito por ST. Está disponible en una carcasa LGA (HLGA) completamente encapsulada con aberturas. El componente funciona en un rango de temperatura garantizado de -40 ° C a +85 ° C. La carcasa está provista de aberturas para que la presión externa pueda alcanzar el elemento sensor. [21].

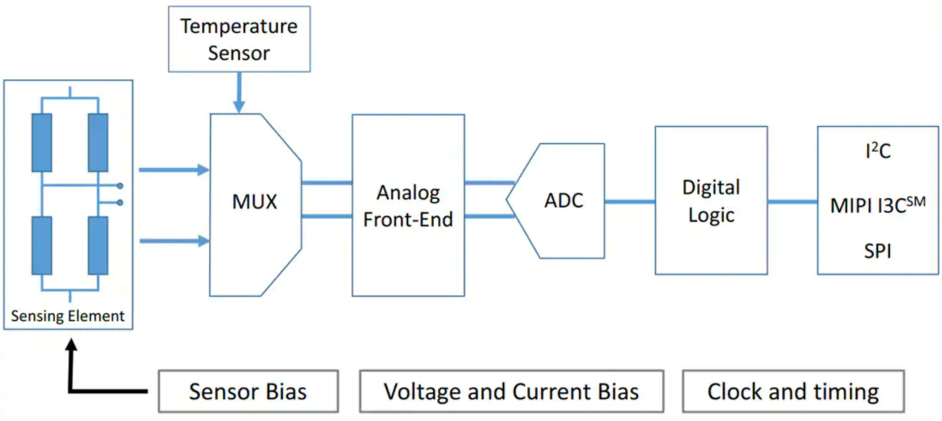


Figure 3.8: Diagrama de bloques del sensor de presión. [28]

Sensor de humedad

El sensor digital capacitivo de humedad y temperatura HTS221 de STMicroelectronics es un sensor ultra-compacto de humedad relativa y temperatura. Incluye un elemento sensor y un circuito integrado de aplicación específica (ASIC) de señal mixta para proporcionar la información de medición a través de interfaces digitales en serie.

El elemento sensor consiste en una estructura de condensador plano dieléctrico de polímero capaz de detectar las variaciones de humedad relativa. El sensor de humedad digital capacitivo HTS221 está disponible en un pequeño encapsulado HLGA que garantiza su funcionamiento en un rango de temperaturas de -40°C a +120°C.

El sensor de temperatura que tiene este dispositivo se puede utilizar como sensor secundario. La diferencia entre los dos sensores de temperatura está en un rango de 2 grados centígrados.

Mis suposiciones son que los dos funcionan perfectamente, salvo que el sensor de temperatura de este dispositivo detecta un valor diferente debido al encapsulado del mismo. [12].

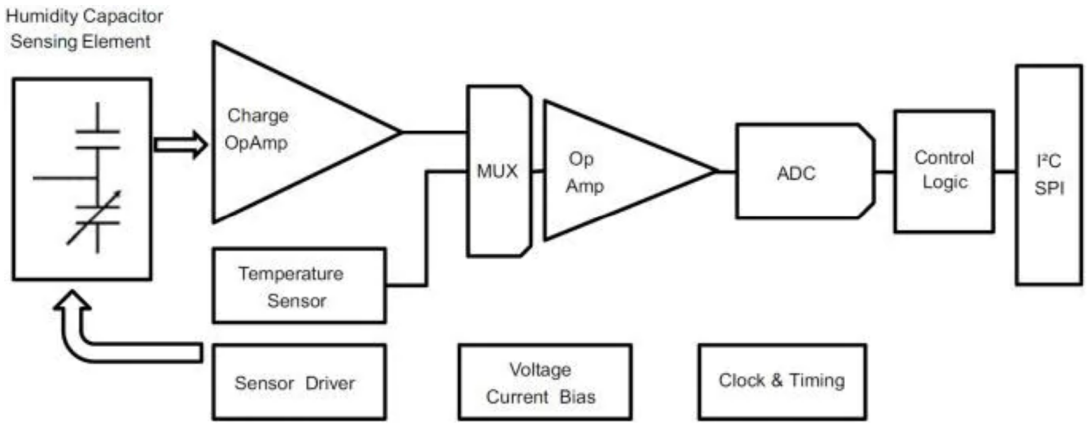


Figure 3.9: Diagrama de bloques del sensor de humedad. [28]

Micrófono

El micrófono de SensorTile.Box (MP23ABS1) es un dispositivo de bajo consumo construido con un elemento sensor capacitivo y una interfaz IC. El elemento sensor, capaz de detectar ondas acústicas, se fabrica mediante un proceso de micromecanizado de silicio especializado para producir sensores de audio. Tiene un punto de sobrecarga acústica de 130 dBSPL con una relación señal/ruido típica de 64 dB [20]. La sensibilidad del MP23ABS1 es de -38 dBV ±1 dB @ 94 dBSPL, 1 kHz. El MP23ABS1 está disponible en un encapsulado compatible con la soldadura por reflujo y está garantizado para funcionar en un amplio rango de temperaturas de -40 °C a +85 °C.

Módulo Bluetooth

El dispositivo también dispone de un módulo Bluetooth de bajo consumo (BlueNRG-M2). La plataforma de evaluación EVAL-SPBTLE-1S se basa en el módulo proce-

sador de aplicaciones de muy bajo consumo SPBTLE-1S, que cumple con las especificaciones Bluetooth Low Energy v4.2 y admite las funciones de maestro, esclavo y maestro y esclavo simultáneos. El módulo se basa en el sistema en chip BlueNRG-2 y toda la pila y los protocolos de Bluetooth de baja energía están integrados en el módulo. El módulo BlueNRG-M2 proporciona una plataforma de RF completa en un factor de forma diminuto [37].

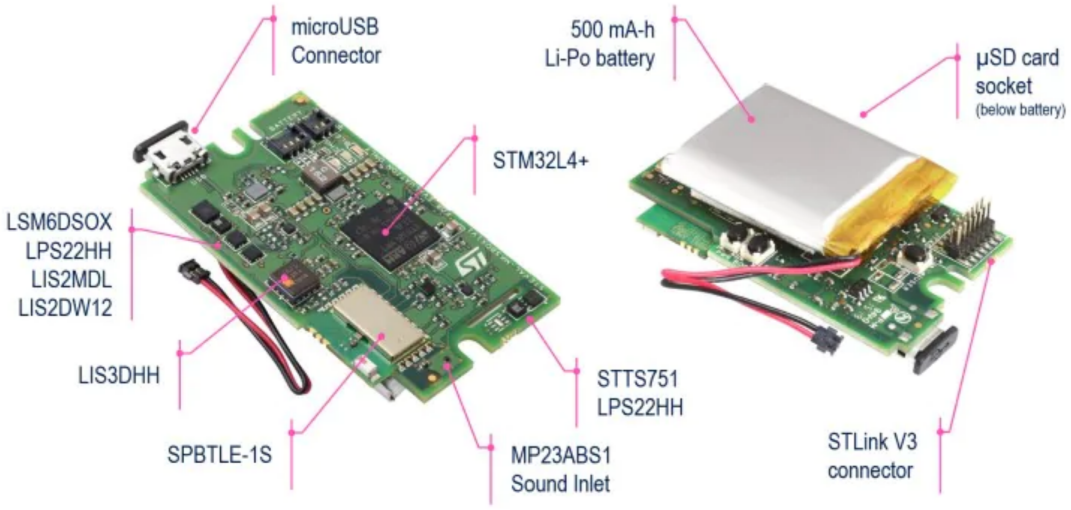


Figure 3.10: Esquema general del dispositivo y sus sensores [19]

3.3 Núcleo de aprendizaje automático

El núcleo de aprendizaje automático disponible para el dispositivo SensorTile.Box (LSM6DSOX), es una de las principales funciones integradas. Está compuesto por un conjunto de parámetros configurables y árboles de decisión capaces de implementar algoritmos en el propio sensor.

Los algoritmos adecuados para el núcleo de aprendizaje automático son aquellos que pueden implementarse siguiendo un enfoque inductivo, que implica la búsqueda de patrones a partir de las observaciones. Algunos ejemplos de algoritmos que siguen este enfoque son: el reconocimiento de movimientos, el reconocimiento de la actividad física, la detección de la de vibraciones, es decir, principalmente el reconocimiento de patrones de datos detectables con los sensores de este dispositivo bien sea patrones de movimiento o patrones de actividad atmosférica.

La idea detrás del núcleo de aprendizaje automático es utilizar el acelerómetro, el giroscopio y los datos de los sensores externos (legibles a través de la interfaz maestra I^2C) para calcular un conjunto de parámetros estadísticos seleccionables por el usuario (la media, la varianza, el pico a pico, etc.) en una ventana de tiempo definida. Este módulo también dispone de filtros que son aplicables a los valores de entrada (datos recolectados con los sensores del dispositivo) que a su vez, la salida de estos filtros se puede utilizar como parámetros de entrada para la generación de un árbol de decisión que se puede almacenar en el dispositivo.

El árbol de decisión es un árbol binario compuesto por una serie de nodos. En cada nodo, se evalúa un parámetro estadístico frente a un umbral para establecer la evolución en el siguiente nodo. Cuando se alcanza una hoja (uno de los últimos nodos del árbol), el árbol de decisión genera un resultado que se puede leer a través de un registro de memoria dedicada del dispositivo.

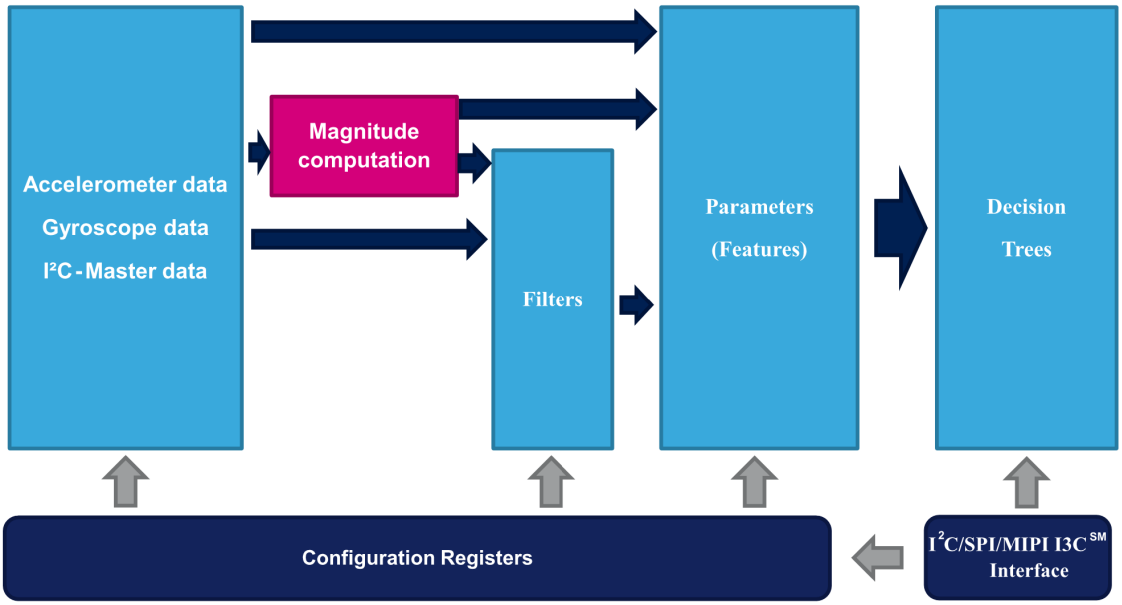


Figure 3.11: Esquema general del núcleo de aprendizaje automático [24].

El núcleo de aprendizaje puede ser configurado para funcionar a cuatro tasas diferentes de frecuencia de salida (12.5 Hz, 26 Hz, 52 Hz y 104 Hz). Esto es muy útil a la hora de generar el árbol de decisión dado que permite configurar la precisión del dispositivo al detectar los patrones de datos.

Para implementar la capacidad de procesamiento de aprendizaje automático del LSM6DSOX, es necesario emplear el método del aprendizaje supervisado. Consiste en la identificación de los patrones de datos que se intentan reconocer con el dispositivo, la recolección de una variedad de colecciones de datos para cada patrón de datos y el análisis de los datos recogidos para poder aprender una condición genérica para el árbol de decisión.

Si se intenta utilizar el dispositivo para el reconocimiento de la actividad física de las personas, por ejemplo, se deberían de recolectar datos en función del estado que se quiera aprender y preferiblemente de una variedad de individuos. Por ejemplo, si nuestra intención es detectar el estado "parado" y "en movimiento" de las personas, habría que recolectar los datos de varios individuos y siempre tener las colecciones de datos clasificadas según los estados a detectar.

Emplear el método de aprendizaje supervisado sirve para definir bien las características que se utilizaran en la detección de los distintos estados. Además, sirve para definir los filtros que se emplearan para mejora de la lectura de los datos y por último ayuda a generar el árbol de decisión que se empleara para la detección de los distintos estados. Una vez generado el árbol de decisión, se puede cargar en el dispositivo utilizando una de las herramientas ofrecidas por el fabricante (AlgoBuilder) y este se ejecutará de una manera óptima, minimizando el consumo del dispositivo.

El núcleo de aprendizaje automático se compone principalmente de tres bloques:

Datos del sensor, Bloque Computacional y Árbol de decisión. El primer bloque está compuesto por datos procedentes del acelerómetro y el giroscopio (que están incorporados en el dispositivo), o de un sensor externo adicional que puede conectarse a LSM6DSOX a través de la interfaz maestra I²C (hub del sensor).

Las entradas del núcleo de aprendizaje automático definidas en el primer bloque se utilizan en el segundo bloque (Bloque Computacional), donde se pueden aplicar filtros y parámetros estadísticos calculados a partir de los parámetros de entrada (o de los datos filtrados) en una ventana de tiempo definida, seleccionable por el usuario.

Los parámetros calculados en el Bloque Computacional se utilizarán como entrada para el tercer bloque del núcleo de aprendizaje automático. Este bloque (Árbol de decisión), incluye el árbol binario que evalúa los parámetros estadísticos calculados a partir de los datos de entrada. Los resultados del árbol de decisión también pueden ser filtrados por un filtro opcional llamado "Meta-clasificador". Los resultados del núcleo de aprendizaje automático serán los resultados del árbol de decisión que incluyen opcionalmente el meta-clasificador [24].

Entradas

El módulo LSM6DSOX funciona como un sensor combinado (acelerómetro + giroscopio), generando datos de salida de aceleración y velocidad angular. Los datos de 3 ejes de la aceleración y la tasa angular pueden utilizarse como entrada para el núcleo de aprendizaje automático. Es importante mencionar que la velocidad de los datos de entrada debe ser igual o superior a la velocidad de datos del núcleo de aprendizaje automático.

Dado que es posible conectar un sensor externo (por ejemplo, un magnetómetro) al LSM6DSOX a través de la interfaz maestra I²C (hub del sensor), los datos procedentes de un sensor externo también pueden utilizarse como entrada para el procesamiento de aprendizaje automático. En este caso, los primeros seis bytes del hub del sensor (dos bytes por eje) se consideran como entrada para el Machine Learning Core (MLC) o núcleo de aprendizaje automático. Cuando se utiliza un sensor externo, la sensibilidad del sensor externo debe configurarse a través de los registros del propio sensor.

Filtros

Los datos de entrada vistos en la sección anterior pueden ser filtrados por diferentes tipos de filtros disponibles en la lógica del Machine Learning Core (MLC). El elemento básico de filtrado del núcleo de aprendizaje automático es un filtro IIR de segundo orden. En el procesado de señales, un filtro digital bicuadrado es un filtro lineal recursivo de segundo orden, que contiene dos polos y dos ceros. Es importante mencionar que los filtros disponibles en el bloque MLC son independientes de cualquier otro filtro disponible en el dispositivo. La gran variedad de filtros disponibles para el bloque MLC se puede encontrar en la documentación del dispositivo.

Ecuación del filtro lineal recursivo de segundo orden:

$$H(Z) = \frac{b_1 + b_{2Z}^{-1} + b_{3Z}^{-2}}{1 + a_{2Z}^{-1} + a_{3Z}^{-2}}$$

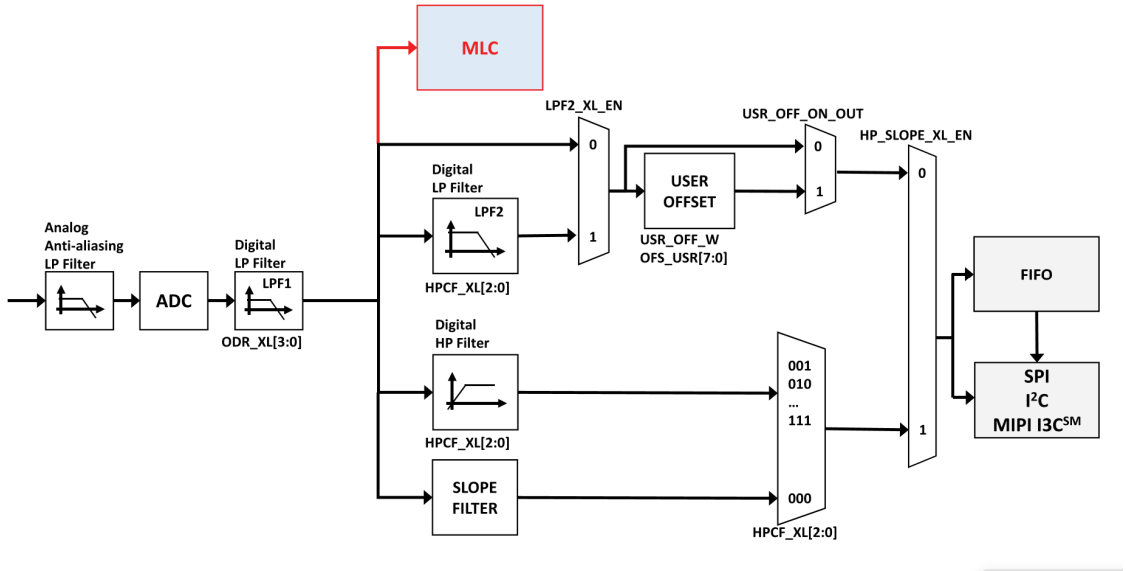


Figure 3.12: Esquema de la entrada de MLC (acelerómetro) [24].

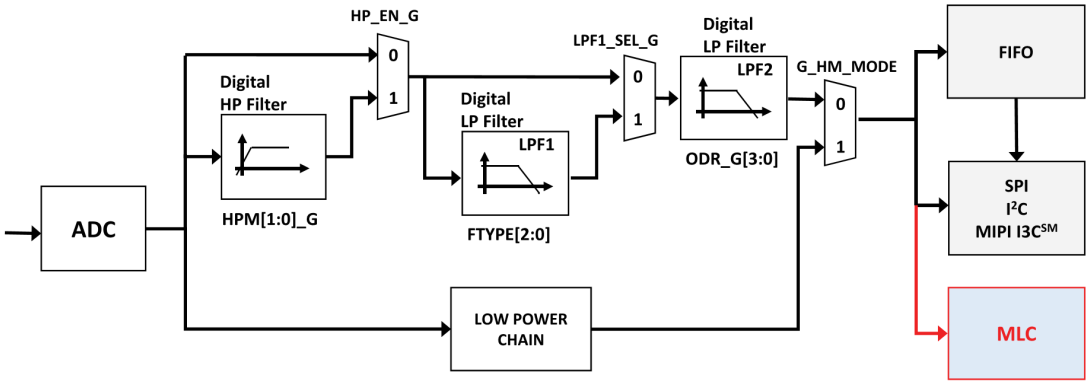


Figure 3.13: Esquema de la entrada de MLC (giróscopo) [24].

Parámetros estadísticos

Los datos calculados a partir de las entradas del núcleo de aprendizaje automático se denominan parámetros estadísticos. Todos los parámetros se calculan dentro de una ventana de tiempo definida. El tamaño de la ventana tiene que ser determinado por el usuario y es muy importante para el procesamiento del aprendizaje automático, ya que todos los parámetros estadísticos del árbol de decisión se evaluarán en esta ventana de tiempo. No es una ventana móvil, las características se calculan una sola vez por cada muestra WL (donde WL es el tamaño de la ventana).

La longitud de la ventana puede tener valores de 1 a 255 muestras. La elección del valor de la longitud de la ventana depende de la tasa de datos del sensor, que introduce una latencia para la generación del resultado del núcleo de aprendizaje automático y de la aplicación o algoritmo específico. En un algoritmo de reconocimiento de actividad, por ejemplo, se puede decidir computar las características cada 2 o 3 segundos, lo que significa que considerando que los sensores funcionan a 26 Hz, la longitud de la ventana debe ser de 50 o 75 muestras respectivamente.

A continuación, se representa un listado de los posibles parámetros estadísticos

que se pueden calcular con el bloque MLC.

1. Media: cálculo del promedio del valor de entrada para un tamaño de ventana predefinido.
2. Varianza: cálculo de la desviación del valor de entrada respecto a un tamaño de ventana predefinido.
3. Energía: cálculo de la energía de la entrada respecto al tamaño de la ventana.
4. Valor pico-a-pico
5. Paso por cero: calcula el número de veces que la entrada seleccionada cruza un determinado umbral. Este umbral interno se define como la suma entre el valor medio calculado en la ventana anterior y la histéresis definida por el usuario.

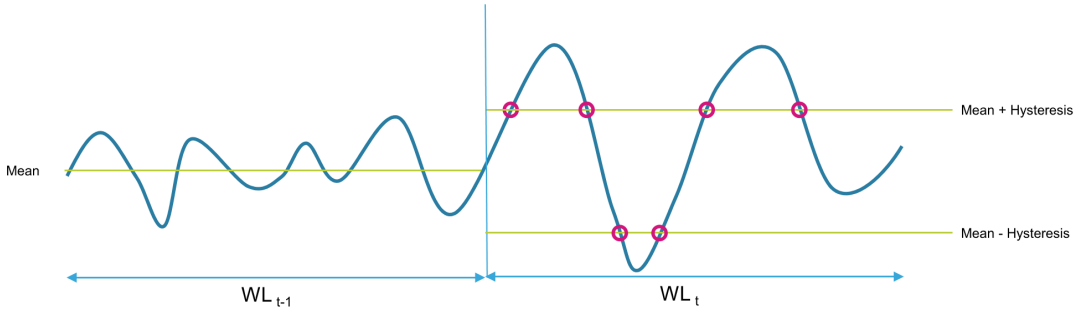


Figure 3.14: Paso por cero [24].

6. Paso por cero positivo: idéntico a la función paso por cero, sólo las transiciones con pendientes positivas se consideran para este cálculo.
7. Paso por cero negativo: idéntico a la función paso por cero, sólo las transiciones con pendientes negativas se consideran para este cálculo.
8. Detector de picos: cuenta el número de picos (positivos y negativos) de la entrada seleccionada en la ventana de tiempo definida. El usuario debe definir un umbral para esta característica, y se considera un buffer de tres valores para la evaluación. Si el segundo valor del buffer de tres valores es mayor (o menor) que los otros dos valores de un umbral seleccionado, el número de picos aumenta. El buffer de tres valores considerado para el cálculo de esta característica es un buffer móvil dentro de la ventana de tiempo.
9. Detector de picos positivo: cuenta el número de picos positivos de la entrada seleccionada en la ventana de tiempo definida definido.
10. Detector de picos negativo: cuenta el número de picos negativos de la entrada seleccionada en la ventana de tiempo definida definido.
11. Valor mínimo: calcula el mínimo valor recibido en la ventana de datos predefinida.
12. Valor máximo: calcula el máximo valor recibido en la ventana de datos predefinida.

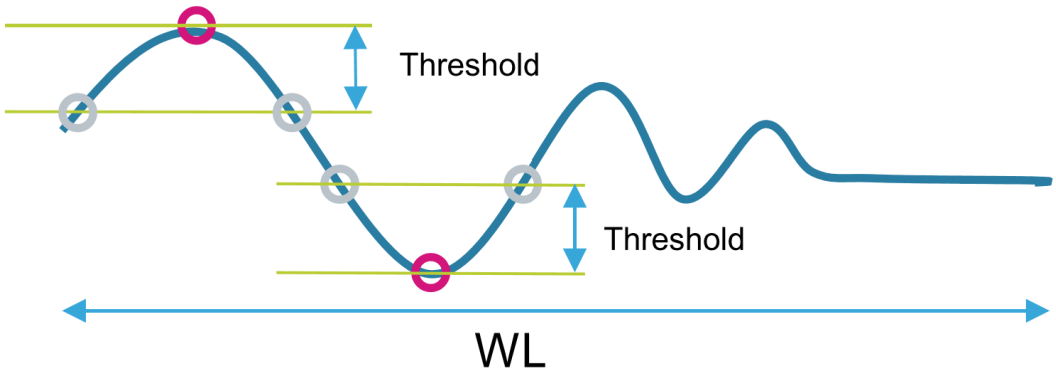


Figure 3.15: Detector de picos [24].

Árbol de decisión

El árbol de decisión es el modelo de predicción construido a partir de los datos de entrenamiento que se pueden almacenar en el LSM6DSOX. Los datos de entrenamiento son los registros de datos adquiridos para cada clase que se va a reconocer. Los resultados de los bloques de cálculo descritos en las secciones anteriores son las entradas del árbol de decisión. Cada nodo del árbol de decisión contiene una condición, en la que una característica se evalúa con un determinado umbral. Si la condición es verdadera, se evalúa el siguiente nodo de la ruta verdadera. Si la condición es falsa, se evalúa el siguiente nodo de la ruta falsa. El estado del árbol de decisión evolucionará nodo a nodo hasta que se encuentre un resultado. El resultado del árbol de decisión es una de las clases definidas al principio de la recogida de datos. El árbol de decisión genera un nuevo resultado en cada ventana de tiempo. Los resultados del árbol de decisión también pueden ser filtrados por un filtro adicional (opcional) llamado "Meta-clasificador".

El meta-clasificador utiliza algunos contadores internos para filtrar las salidas del árbol de decisión. Las salidas del árbol de decisión pueden dividirse en subgrupos (por ejemplo, las clases similares pueden gestionarse en el mismo subgrupo). Se dispone de un contador interno para todos los subgrupos de las salidas del árbol de decisión. El contador del subgrupo específico se incrementa cuando el resultado del árbol de decisión es una de las clases del subgrupo y se disminuye en caso contrario. Cuando el contador alcanza un valor definido, que se llama "contador final" (establecido por el usuario), la salida del núcleo de aprendizaje automático se actualiza. Los valores permitidos para los contadores finales son de 0 a 14.

3.4 Firmware

El dispositivo SensorTile.Box a pesar de ser un kit funcional para la recolección de datos, ofrece la posibilidad de ser programado completamente. El código del firmware de este dispositivo está publicado en la red por la empresa de fabricación y esta ofrece varios ejemplos para ser modificados y utilizados en la recolección de datos.

En los sistemas electrónicos y la informática, el firmware es un componente electrónico tangible con instrucciones de software integradas, como una BIOS. Normalmente, esas instrucciones de software se utilizan para indicar a un dispositivo

electrónico cómo debe funcionar. Ejemplos típicos de dispositivos que contienen firmware son los sistemas integrados (como los semáforos, los aparatos de consumo y los relojes digitales), los ordenadores, los periféricos informáticos, los teléfonos móviles y las cámaras digitales. El firmware que contienen estos dispositivos proporciona el programa de control del aparato.

El firmware se guarda en dispositivos de memoria no volátil, como la ROM, la EPROM o la memoria flash. El cambio del firmware de un dispositivo puede realizarse raramente o nunca durante su vida económica; algunos dispositivos de memoria de firmware están instalados permanentemente y no pueden cambiarse después de su fabricación. Las razones más comunes para actualizar el firmware incluyen la corrección de errores o la adición de características al dispositivo. Esto puede requerir la sustitución física de los circuitos integrados de la ROM o la reprogramación de la memoria flash mediante un procedimiento especial. El firmware, como la ROM BIOS de un ordenador personal, puede contener sólo las funciones básicas elementales de un dispositivo y proporcionar únicamente servicios al software de nivel superior. El firmware, como el programa de un sistema integrado, puede ser el único programa que se ejecute en el sistema y proporcione todas sus funciones.

Sensing1

Un ejemplo de firmware que el fabricante sugiere a los usuarios de SensorTile Box es 'AI-SENSING1'. Esta versión de firmware ofrece la posibilidad de activar los sensores que uno desee a la frecuencia que desee para la recopilación de datos en la memoria interna del dispositivo. Una función igual de interesante es la posibilidad de poder guardar los datos obtenidos en la red a través del protocolo MQTT. Para el funcionamiento correcto es necesario que en todo momento el dispositivo móvil esté conectado con SensorTile.Box. Esto puede ser algo poco práctico si se decide utilizar el dispositivo para la recolección de datos en animales por ejemplo u otro propósito en el que es necesario recolectar una gran cantidad de datos durante un tiempo prolongado.

Para obtener los datos una vez guardados en la memoria interna, se puede extraer la memoria micro SD del dispositivo y conectarla de manera directa a un ordenador. Sin embargo, hay una manera un poco más segura, aunque algo más tediosa para copiar los datos. Se trata de instalar el firmware que permite conectar el dispositivo a un ordenador a través de un cable micro-usb. Esta conexión crea en el ordenador un disco extraíble en el que son visibles todos los archivos que el dispositivo fue capaz de crear al recolectar los datos.

Este firmware que es ofrecido por el fabricante como una referencia, al arrancar ejecuta la función principal del sistema. Dicha función a su vez hace una llamada a otra función que inicia un proceso en cadena para la configuración del reloj interno del dispositivo, el control de energía y comprueba el modelo del dispositivo. El inicio del hardware también se encarga de comprobar el funcionamiento del módulo Bluetooth, si hay meta-datos guardados y la disponibilidad de los periféricos del mismo.

Una vez hecho el inicio del hardware, la función principal crea tres hilos de trabajo para la gestión de tareas. Los tres hilos disponen de una asignación de prioridad diferente. El hilo con más prioridad se denomina "HostThread". Es el primero encargado de ejecutar la recepción de los mensajes de la conexión Blue-

tooth. Los dos hilos restantes, "ProcessThread" y "UARTConsoleThread" tienen asignada una prioridad normal, lo que significa que empezarán a ejecutarse bajo algunas condiciones específicas que no entran en conflicto con estos hilos. Cabe destacar que el hilo "UARTConsoleThread" se crea solo bajo la condición de uso del firmware 'AI-SENSING1'. Este hilo es el encargado de recibir los comandos por consola del usuario que los envía por medio de la aplicación Android de un smartphone. Al crearse "UARTConsoleThread" el firmware empieza a cargar la lista de comandos que admite esta versión del firmware. Estos comandos suelen variar según las funcionalidades programadas del dispositivo.

También, la función principal crea tres semáforos que se utilizan en la ejecución del código con el propósito de control de flujo de los datos. Los semáforos limitan la ejecución de un código o procedimiento a uno o varios hilos con el fin de evitar alterar los datos o perderlos en un proceso de ejecución. El proceso principal, "main", por último, asigna memoria para el guardado de los mensajes de Bluetooth en una cola que se ejecutaría de manera cronológica. Funciona con el principio 'First in - First out' (FIFO), es decir, el primer mensaje que se guarda en la cola, es el primero en procesarse una vez el sistema esté disponible para ello. Cabe mencionar que la memoria asignada para esta cola de mensajes es dinámica, por lo tanto su tamaño puede variar según el flujo de datos que se gestione al momento. Por último, el proceso principal configura el dispositivo para el mínimo consumo según el modelo en el que se esté ejecutando el software.

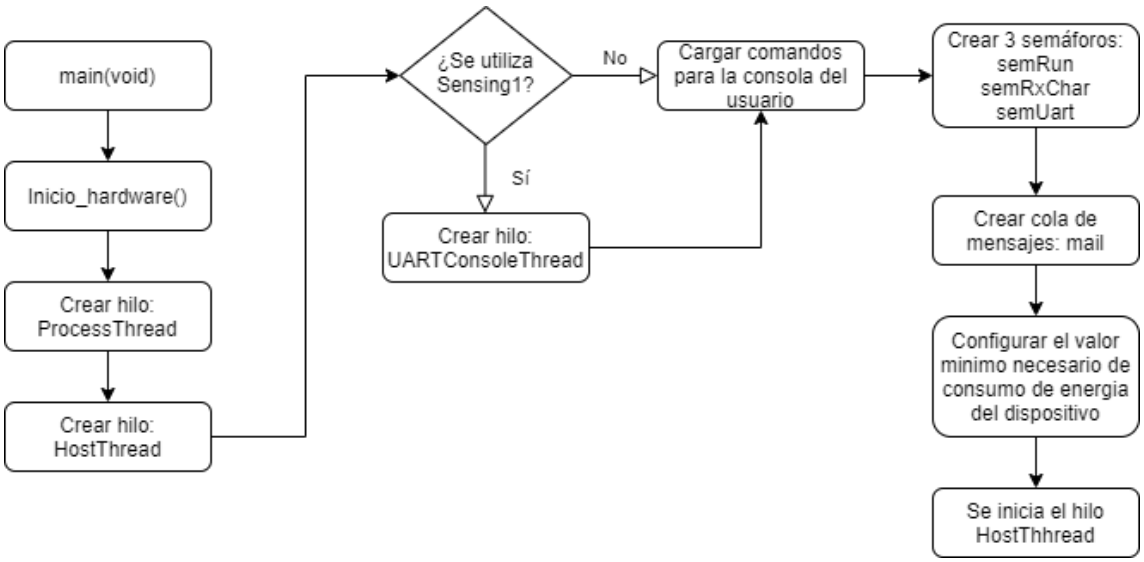


Figure 3.16: Proceso de ejecución de la función principal del firmare (main)

Al arrancar el hilo "HostThread" se crea un bucle infinito que encapsula todos los métodos y funciones que comprende. Antes de procesar nada, el hilo comprueba si hay algún mensaje (mail) recibido en el dispositivo. Si no hay ningún mensaje recibido, el dispositivo sigue esperando en el bucle infinito, en caso contrario procesa el mensaje y activa un proceso posterior según el tipo de mensaje. Cabe mencionar que este hilo al activarse, ocupa el semáforo de control "semRun" que es el encargado de la ejecución de ambos hilos evitando las posibles colisiones a la hora de utilizar los mismos datos.

El primer mensaje "SET_CONECTABLE" es un mensaje preconfigurado del dispositivo para activar el estado de espera de conexión Bluetooth. En este estado

el dispositivo carga una configuración por defecto del módulo Bluetooth y activa un led en modo intermitente para indicar que está preparado para una conexión con un dispositivo móvil. Algunos de los mensajes que se reciben por el dispositivo posteriormente a la conexión son para la actualización del estado de las notificaciones de la aplicación Android y el estado en del dispositivo. Como la aplicación para el teléfono está diseñada para mostrar los valores de algunos sensores por pantalla en un gráfico, es de esperar que algunos de los mensajes Bluetooth que puede recibir el dispositivo en el hilo "HostThread" están destinados a pedir los valores directamente a modo de prueba. El teléfono no puede guardar los datos recibidos de esta manera por el dispositivo, dado que las funciones que activan los procesos de envío de esos datos utilizan una memoria limitada del teléfono para demostrar el funcionamiento y capturar datos.

Otros tipos de mensajes como "BATTERY_INFO" envían a la aplicación movil el porcentaje de carga que tiene la batería interna del dispositivo. El mensaje más relevante que este hilo puede recibir es "SD_CARD_LOGGING" que en este caso, al recibirlo el hilo "HostThread" libera el semáforo de control que ocupa, semRun, para que el siguiente hilo (ProcessThread) pueda encargarse del proceso.

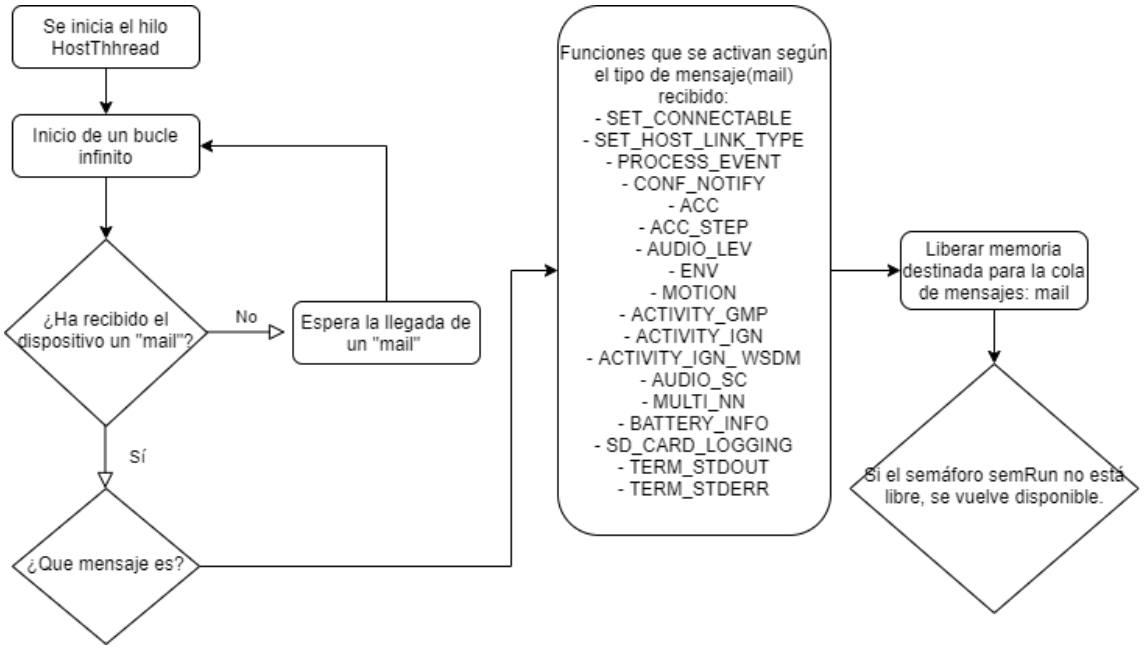


Figure 3.17: Proceso de ejecución del hilo de mayor prioridad del firmware (HostThread)

El hilo "ProcessThread" se activa solo bajo la condición de que el hilo "HostThread" libere el semáforo "semRun". Al igual que el hilo anterior, este crea un bucle infinito en el que comprueba si el semáforo está disponible. Si es así, el hilo bloquea la disponibilidad del semáforo y comprueba los meta-datos de la conexión Bluetooth. Esto es necesario para conseguir sincronizar el reloj del dispositivo con el reloj del teléfono móvil. De esta manera, los datos capturados posteriormente tendrán una marca de tiempo lo más precisa posible.

El hilo al igual que en el caso previo, procesa el mensaje recibido y se dispone a enviar los datos recogidos, que esta vez se envían al móvil para un guardado posterior en una base de datos de la red. Es decir, este hilo se encarga principalmente de la

obtención de los datos. Si es necesario guardar los datos en la red en una base de dados, este proceso es gestionado por este hilo. Esto para con aproximadamente todos los mensajes que es capaz de procesar este hilo, menos con los mensajes de "DATALOG_USE_". Estas últimas dos opciones se activan en el caso de que el dispositivo funcione en el modo de recolección y guardado de datos en la memoria interna del mismo.

Para las opciones de recolección de datos y guardado de estos se activan en cadena varias funciones que controlan el proceso. Para empezar, hay una diferencia entre hacer una grabación de solo audio o hacer una captura simultánea de datos y audio. En el primer caso, haría falta abrir un archivo audio con una extensión "wav" para guardar las muestras de sonido. En el segundo caso haría falta además abrir un archivo con una extensión "CSV" para guardar los datos de los demás sensores. Una vez que estos archivos se abren para guardar los datos, el dispositivo se dispone a capturar y salvar las muestras en dichos ficheros. Este proceso es continuo hasta que el usuario en el dispositivo móvil no lo para. Al parar el proceso de "datalog" el firmware se dispone a cerrar los ficheros de datos, bien sea uno o los dos, según se dé el caso.

Es importante mencionar que las funciones de gestión de estos procesos implican un control en detalle del estado del dispositivo en todo momento. Por ejemplo, antes de captar cualquier dato, el firmware comprueba si hay una conexión Bluetooth. Además comprueba si hay una conexión con la memoria interna del dispositivo antes de guardar cualquier dato o abrir y cerrar ficheros. Este tipo de comprobaciones se utilizan para condicionar la actividad del dispositivo, ya que si ya existen un par de ficheros abiertos que guardan datos, los datos seguirán guardándose en los mismos hasta recibir nuevas "instrucciones".

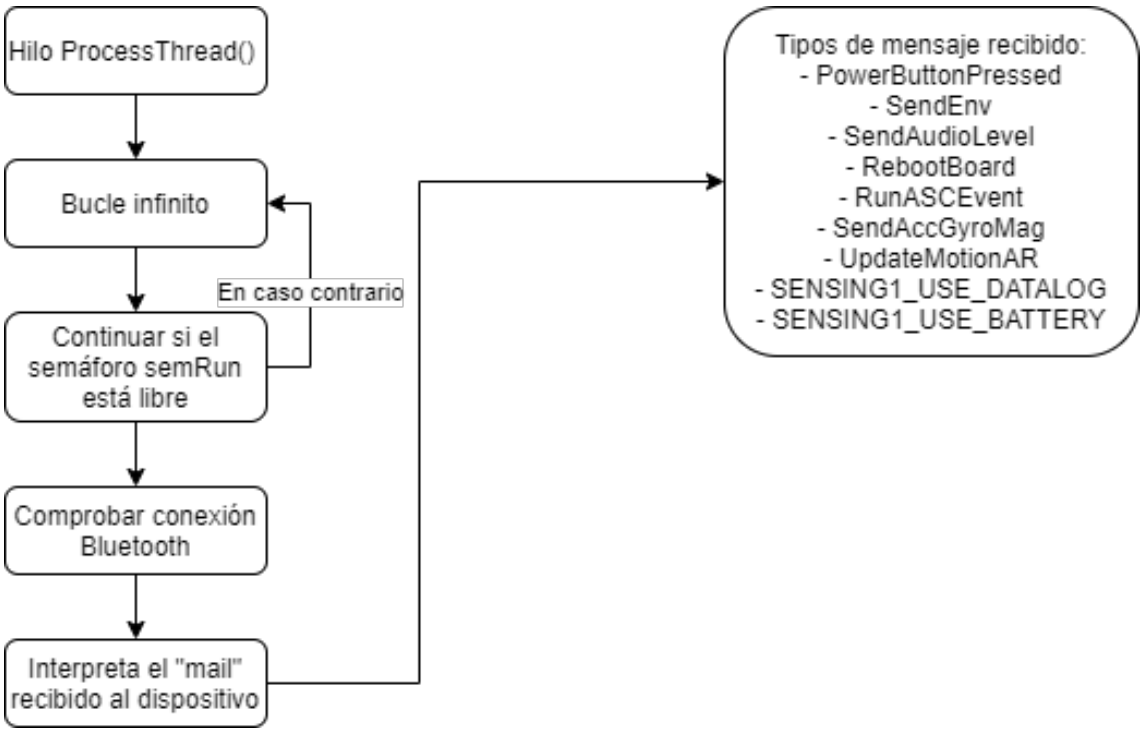


Figure 3.18: Proceso de ejecución del hilo ProcessThread

Otros ejemplos de firmware ofrecen opciones diferentes como aplicaciones de-

mostrativas con inteligencia artificial o maneras de recolección de datos sin la necesidad de estar constantemente conectado con un dispositivo móvil. La versión de firmware 'SNS-ALLMEMS1' ofrece un factor de independencia en el funcionamiento que es muy útil a la hora de utilizarlo para la recolección de datos sin tener que monitorizar el dispositivo de manera visual constantemente. El punto negativo de esta versión de firmware está en la frecuencia máxima de recolección de datos que el dispositivo es capaz de hacer. La funcionalidad offline de este dispositivo solo permite como mucho recoger datos de los sensores indicados cada segundo. Como es de esperar, una frecuencia tan baja en la recolección de datos es insuficiente para la monitorización de animales o implementaciones que requieran de una sensibilidad mayor para poder captar los cambios de valores.

3.5 Cloud Server

En este apartado se expone la infraestructura que se ha configurado con el fin de poder utilizar el dispositivo SensorTile.Box para la recopilación de datos. Esta infraestructura permite guardar los datos obtenidos del dispositivo en una base de datos en la red para un procesado posterior.

La infraestructura que se ha utilizado está compuesta de varios componentes clave. Se ha utilizado un broker MQTT para reenviar los datos del dispositivo móvil a través de la red a la base de datos. Para la base de datos de ha utilizado un servidor APACHE y la base de datos MySQL como gestor para guardado de los datos. Los dos componentes, tanto el servidor APACHE como la base de datos MySQL forman parte de una aplicación conjunta llamado XAMPP.

El broker MQTT utilizado se denomina Mosquitto [17]. Eclipse Mosquitto es un broker de mensajes de código abierto que implementa las versiones 5.0, 3.1.1 y 3.1 del protocolo MQTT. Mosquitto es ligero y se puede utilizar en todos los dispositivos, desde ordenadores de placa única de baja potencia hasta servidores completos. La función del broker es de recibir los datos enviados por la red del dispositivo móvil y reenviarlos a un servidor que tiene activa una base de datos.

XAMPP es un paquete de soluciones de servidor web multiplataforma, gratuito y de código abierto, desarrollado por Apache Friends [3], que consiste principalmente en el servidor HTTP Apache, la base de datos MariaDB y los intérpretes para scripts escritos en los lenguajes de programación PHP y Perl. Dado que la mayoría de los despliegues de servidores web reales utilizan los mismos componentes que XAMPP, hace posible la transición de un servidor de pruebas local a un servidor en vivo. Esta aplicación facilita el uso de un servidor APACHE y una base de datos MySQL, dado que estos componentes vienen preinstalados y configurados con la aplicación. En consecuencia esto permite el arranque de los servicios ofrecidos con solo un botón.

El servicio en red se configuró utilizando un servidor APACHE [9]. Desarrollado y mantenido por Apache Software Foundation. APACHE es también un software de código abierto disponible de forma gratuita. Es rápido, fiable y seguro. Puede ser altamente personalizado para satisfacer las necesidades de muchos entornos diferentes mediante el uso de extensiones y módulos.

Un servidor web es un software informático y un hardware subyacente que acepta peticiones a través de HTTP, el protocolo de red creado para distribuir páginas web [32], o su variante segura HTTPS. Un agente de usuario, normalmente un navegador o un rastreador web, inicia la comunicación haciendo una petición de un recurso

específico mediante HTTP, y el servidor responde con el contenido de ese recurso o con un mensaje de error. El servidor también puede aceptar y almacenar recursos enviados por el agente de usuario si está configurado para ello.

Un servidor puede ser un solo ordenador, o incluso un sistema integrado como un router con una interfaz de configuración incorporada, pero los sitios web de alto tráfico suelen ejecutar servidores web en flotas de ordenadores diseñados para manejar un gran número de solicitudes de documentos, archivos multimedia y scripts interactivos. Un recurso enviado desde un servidor web puede ser un archivo preexistente disponible en el servidor, o puede ser generado en el momento de la solicitud por otro programa que se comunica con el programa servidor. El primero suele ser más rápido y se almacena más fácilmente en la caché para las solicitudes repetidas, mientras que el segundo admite una gama más amplia de aplicaciones. Los sitios web que sirven contenidos generados suelen incorporar archivos almacenados siempre que es posible.

Como sistema gestor de la base de datos se ha utilizado MySQL [13]. Una base de datos relacional organiza los datos en una o varias tablas en las que los tipos de datos pueden estar relacionados entre sí; estas relaciones ayudan a estructurar los datos. SQL es un lenguaje que los programadores utilizan para crear, modificar y extraer datos de la base de datos relacional, así como para controlar el acceso de los usuarios a la base de datos. Además de las bases de datos relacionales y SQL, MySQL funciona como sistema operativo para implementar una base de datos relacional en el sistema de almacenamiento de un ordenador, gestiona los usuarios, permite el acceso a la red y facilita la comprobación de la integridad de la base de datos y la creación de copias de seguridad.

Como se puede ver el en esquema mostrado 3.19, el primer paso sería instalar en el ordenador broker MQTT (Mosquitto). Una vez instalado, lo conveniente sería configurarlo para asegurarnos de que hay una seguridad mínima de acceso al mismo, es decir, se debería de crear un usuario y contraseña. Esto claramente no ofrecerá mucha protección, pero es útil para tener un control de acceso. Después de editar el archivo de configuración de Mosquitto, hay que asegurarse que el servicio está funcionando correctamente. Esto se consigue abriendo una ventana de comandos en el sistema operativo Windows con permiso de administrador e introduciendo el comando "netstat -ab". Este comando devuelve una lista de todos los puertos abiertos del ordenador. Si se observa en dicha lista el puerto 1883 en modo escucha, entonces el broker funciona correctamente, en caso contrario habría que comprobar la configuración de Mosquitto.

Un paso opcional pero muy recomendable es la instalación de la aplicación MQTT Explorer. Este programa funciona como un cliente de acceso al broker. La aplicación ofrece la posibilidad de monitorizar el tráfico en tiempo real del broker MQTT. Al funcionar como cliente del broker, es necesario autenticarse para acceder con el usuario y contraseña definidos en el archivo de configuración de Mosquitto. Cabe destacar que MQTT Explorer funciona como un visualizador de los datos pero no es capaz de guardarlos en el ordenador. Por lo tanto, hace falta guardar los datos en una base de datos online.

El siguiente paso en el proceso de montaje de la infraestructura para nuestro servidor online sería la instalación y configuración de la aplicación XAMPP. En este caso hay que comprobar que en el archivo de configuración de XAMPP está bien definida la ruta para el servidor APACHE y comprobar que este arranca en

la dirección IP local. Una vez terminada la configuración del servidor, se puede arrancar el mismo junto con la base de datos MySQL. La comprobación fácil del funcionamiento de ambos servicios se puede hacer utilizando cualquier motor de búsqueda para acceder al servicio "http://localhost/phpmyadmin". Mediante el uso de esa dirección estamos accediendo a la base de datos que funciona en el servidor APACHE. Es decir, si no se puede acceder a este recurso, se debería comprobar por separado el funcionamiento del servidor, su configuración y el funcionamiento de la base de datos y su configuración.

Si nuestra base de datos funciona perfectamente, podemos crear un usuario y contraseña de acceso para esta. Esto servirá para proteger al usuario de errores innecesarios. Creando un usuario podemos limitar el acceso para la base de datos, es decir, quitarle el privilegio del administrador. Este paso no es necesario pero recomendable ya que más tarde necesitaremos un 'script' para gestionar el guardado y la autenticación. Quitar el permiso de administrador me parece algo sensato en este caso.

A continuación, se crea una base de datos que utilizaremos posteriormente para guardar los datos. Cabe mencionar que la creación del usuario y contraseña, al igual de la base de datos se puede hacer desde el acceso al recurso "http://localhost/phpmyadmin", utilizando código SQL. Si podemos comprobar en el recurso (de manera visual) que existe la base de datos creada con el nombre indicado, podemos disponernos a escribir el código necesario en JavaScript.

El código que necesitamos se va a encargar de hacer una suscripción al broker MQTT. Esta suscripción es necesaria para que Mosquitto sepa que datos de los que recibe tiene que reenviar. La suscripción que se hace será para poder recibir todos los datos que el broker a su vez recibe. El script además de la suscripción se encargará de guardar los datos que recibe del broker en la base de datos online. Es necesario que el código además de las indicaciones mencionadas haga la autenticación con el broker (para poder recibir los datos) y con la base de datos (para poder guardar los datos).

Una vez tengamos el código terminado se guardará en un archivo ".js" y se copiará en la carpeta del servidor web. Para ejecutar el código habría que comprobar primero que los servicios anteriores funcionan perfectamente. Después, en una ventana de comandos buscamos la carpeta que contiene nuestro archivo javascript y ejecutamos "node nombre_archivo.js". Si hemos hecho bien todo el proceso, deberíamos de ser capaces de ver como se guardan los datos accediendo a la base de datos "http://localhost/phpmyadmin". En caso contrario habría que revisar el código, añadir líneas de control y repetir el proceso de ejecución.

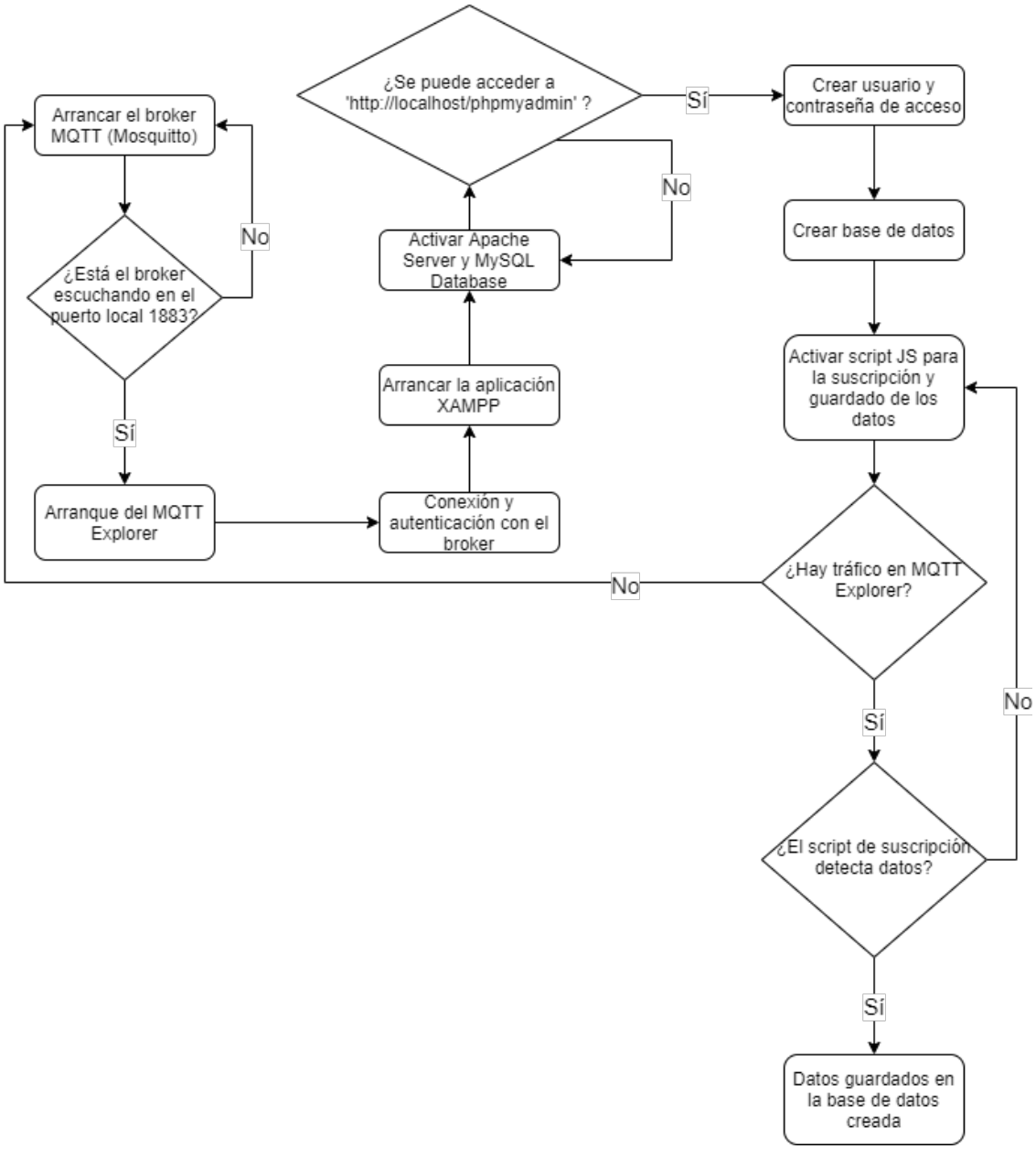


Figure 3.19: Infraestructura del Cloud Server.

	messageID	clientID	topic	message	Enable	DateTime_created
Edit Copy Delete	1	client001	senn/accelerometer/x	-218	1	2021-07-20 07:00:47
Edit Copy Delete	2	client001	senn/accelerometer/y	-64	1	2021-07-20 07:00:47
Edit Copy Delete	3	client001	senn/accelerometer/z	980	1	2021-07-20 07:00:47
Edit Copy Delete	4	client001	senn/temperature/temperature	26.8	1	2021-07-20 07:00:49
Edit Copy Delete	5	client001	senn/gyroscope/x	-0.6	1	2021-07-20 07:00:50
Edit Copy Delete	6	client001	senn/gyroscope/y	0.7	1	2021-07-20 07:00:50
Edit Copy Delete	7	client001	senn/gyroscope/z	-0.2	1	2021-07-20 07:00:50

Figure 3.20: Ejemplo de tabla creada en la base de datos.

Chapter 4

Captura y análisis de datos

4.1 Almacenamiento y captura de datos

En este apartado se explicará la captura de datos con el dispositivo SensorTile.Box a través de dos opciones diferentes. La primera, utilizando una base de datos como servicio en red para guardar los valores generados en el dispositivo. La segunda, utilizando la funcionalidad 'Data Log' del dispositivo para guardar los datos generados en la memoria interna del dispositivo.

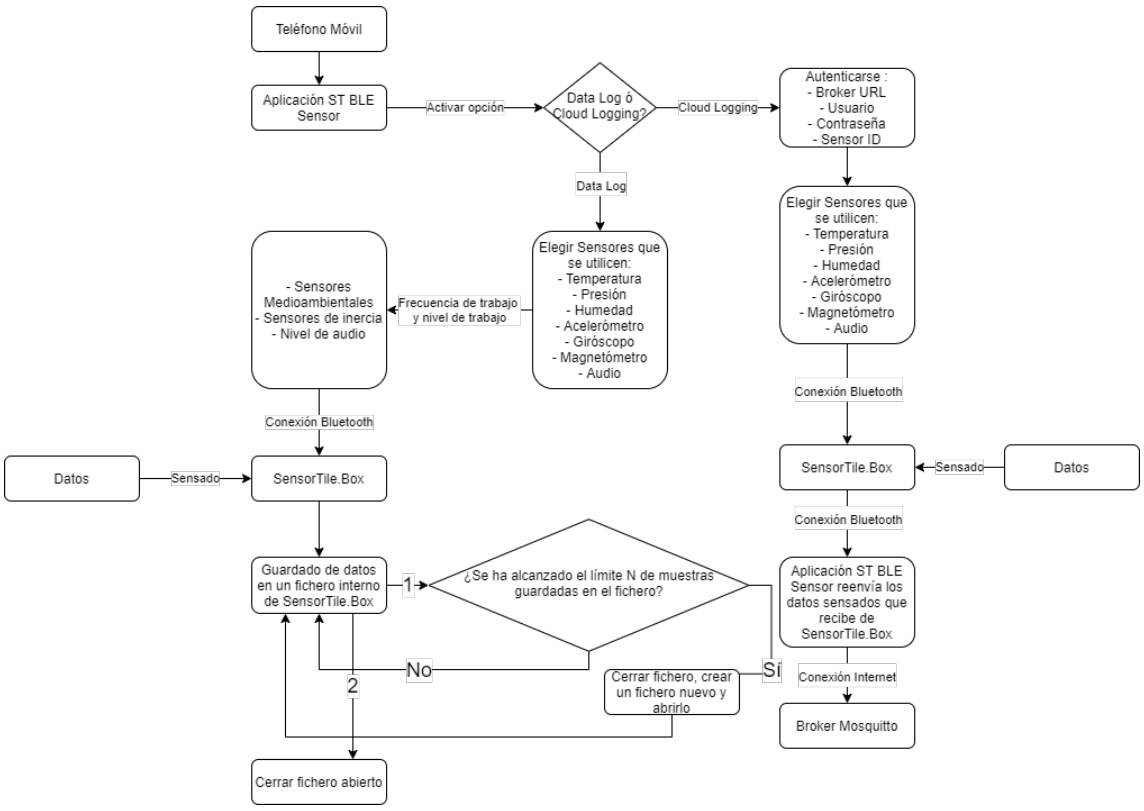


Figure 4.1: Esquema visual del proceso de almacenamiento de los datos.

Para comenzar con la captura de los datos, primero hay que asegurarse que el dispositivo SensorTile.Box está conectado al teléfono móvil mediante la conexión Bluetooth. Esto se consigue con la ayuda de la aplicación móvil ST BLE Sensor, que al abrirla ofrece la posibilidad de buscar posibles dispositivos del fabricante a

los que se puede conectar. Una vez conectado el dispositivo al teléfono móvil, la aplicación nos mostrará por pantalla las funcionalidades que el firmware instalado previamente dispone. En nuestro caso, Sensing1 dispone de las dos funcionalidades, 'Data Log' y guardado de datos en la red.

Al elegir la funcionalidad 'Data Log' en la aplicación móvil, automáticamente se nos ofrecerá la posibilidad de elegir los sensores que estamos dispuestos a utilizar. Además de los sensores que podemos activar, también podemos configurar la frecuencia de trabajo de los sensores medioambientales, la frecuencia de los sensores de inercia y el nivel de audio para el micrófono integrado. Una vez hecha esta mera configuración previa, podremos comenzar con el almacenamiento de los datos en la memoria interna del dispositivo.

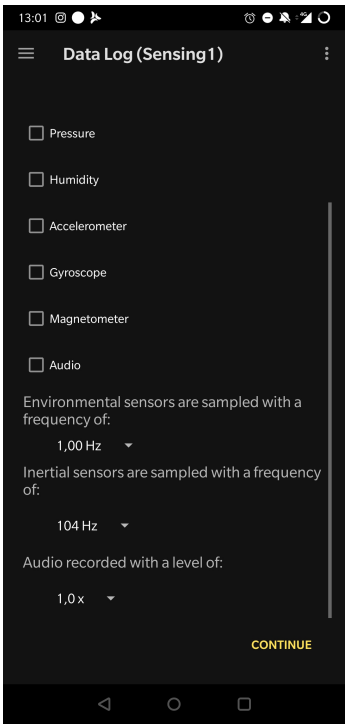


Figure 4.2: Información visual disponible de la configuración del Data Log.

Al pulsar el botón de 'Data Log' en la aplicación, se envía por la conexión Bluetooth la configuración previa que se hizo. Posteriormente, el dispositivo se dispone a crear uno o dos ficheros (dependiendo si se hace un captura de datos con audio) en los que guardará los datos sensados.

Como se ha comentado antes, la función de 'Data Log' depende constantemente de la conexión Bluetooth con la aplicación del dispositivo móvil. En la versión del firmware de ejemplo que ofrece el fabricante, para parar el proceso de 'Data Log' del dispositivo, hay que pulsar un botón en la aplicación móvil. Es decir, este proceso dura de manera indefinida hasta que el usuario no indica lo contrario. Esta funcionalidad es poco efectiva dado que el dispositivo pierde la conexión con la aplicación del teléfono constantemente. Estando en un proceso de 'Data Log' y perder la conexión Bluetooth implica que los datos capturados hasta el momento no se pueden guardar correctamente y se pierden. Esto es debido a que el dispositivo no recibe una orden de parar el proceso y por lo tanto no cierra el fichero con los datos que abre en un principio. Para solucionar este problema, se ha modificado

ligeramente la funcionalidad del firmware. Concretamente, se ha añadido varias líneas de código que cierra el fichero abierto previamente y abre un fichero nuevo para seguir guardando datos cada 'N' muestras.

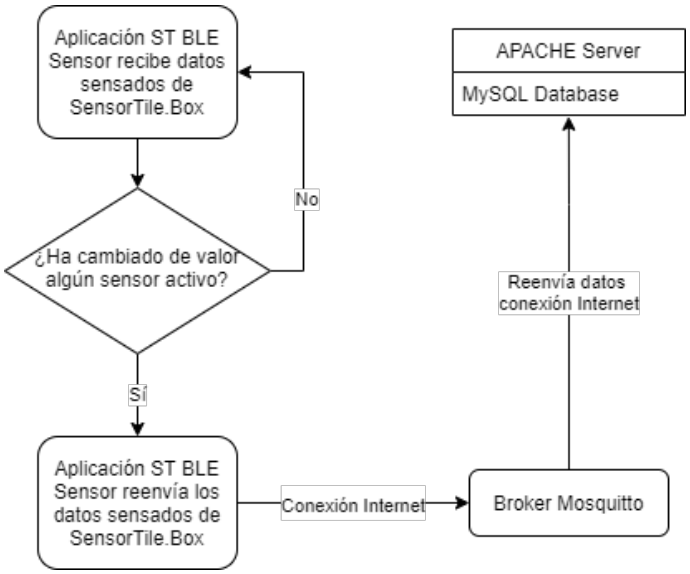


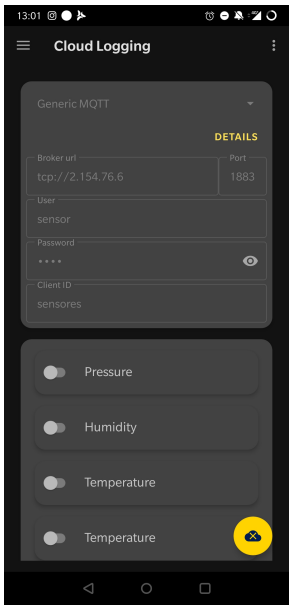
Figure 4.3: Transmisión y guardado de datos en la red.

Entonces, cuando el proceso 'Data Log' se activa, el dispositivo empieza a contar el número de muestras que escribe en el fichero. Si este numero llega a un límite 'N', el dispositivo cierra el fichero y abre uno nuevo para seguir guardando datos. Este proceso sigue en un ciclo hasta que el usuario pare la función de 'Data Log'. De esta manera, si se pierde la conexión en medio del proceso con un fichero abierto, se perderán solo una pequeña parte de los datos recogidos, así pudiendo hacer un guardado de datos más seguro.

Al acceder desde la aplicación móvil a la función de 'Cloud Logging' se nos pedirá autenticarnos. Antes de la autenticación deberemos comprobar que nuestra infraestructura de Cloud Server funciona perfectamente. Posteriormente, utilizando el protocolo TCP copiaremos la dirección IP de nuestro router añadiendo el puerto 1883. Como se ha mencionado, el puerto 1883 del router está redireccionado al puesto local del ordenador con el mismo identificador (1883). Además de la URL del broker, hará falta indicar la identificación del usuario y la contraseña creadas previamente en la configuración de Mosquitto. Por último se puede indicar un identificador del cliente, este es opcional, dado que simplemente se utiliza para que el broker pueda rastrear mejor de que dispositivo recibe los datos (en el caso de que haya más de uno enviando datos al broker). De todas formas, la configuración de Mosquitto tiene prevista la ausencia de un identificador de cliente, y en ese caso el broker asigna uno por defecto.

Una vez hecha la autenticación podemos conectarnos a Mosquitto. Posteriormente podemos elegir los sensores de los cuales estamos interesados en guardar los datos en la red. La activación de los sensores es automática, es decir, basta con indicar que sensores queremos activos y los datos empezarán a enviarse por la red sin esperas. Los datos recogidos en este caso se envían primero por la conexión Bluetooth a la aplicación del teléfono móvil. Estos se comprueban en la aplicación si los valores de los sensores han cambiado con respecto a los previos transmitidos.

(a) Autenticación para el guardado de datos en red.



(b) Información visual disponible de los sensores del dispositivo.

Si los valores han cambiado, se reenvían al broker, en caso contrario se desechan.

Cloud Logging no tiene ninguna opción de regulación de frecuencia de trabajo como 'Data Log' para no saturar los canales de transmisión de datos. En consecuencia, la aplicación solo reenvía al broker los datos que han cambiado con respecto a los valores previos. De esta manera una vez obtenidos en la base de datos, se podrían modificar la colección de datos y añadir más muestras (sí hiciese falta) teniendo en cuenta los tiempos de muestreo de estos. Mosquitto, al recibir los datos, comprueba si tiene alguna suscripción para estos, en caso afirmativo los reenvia al cliente suscrito, que en este caso es el servidor APACHE.

4.2 Generación de configuración

El procesamiento de Machine Learning se obtiene a través de la lógica del árbol de decisión. Un árbol de decisión es una herramienta matemática compuesta por una serie de nodos configurables. Cada nodo se caracteriza por una condición "if-then-else", en la que una señal de entrada (representada por parámetros estadísticos calculados a partir de los datos del sensor) se evalúa frente a un umbral.

Los resultados del árbol de decisión pueden ser leídos por el procesador de la aplicación en cualquier momento. Además, existe la posibilidad de generar una interrupción por cada cambio en el resultado del árbol de decisión.

En este caso se va a crear un árbol de decisión para detectar dos estados de una persona, parado y caminando. Antes de comenzar, haría falta disponer de los datasets necesarios para el entrenamiento de nuestro árbol de decisión. Para este ejemplo, se recogieron los datos con el dispositivo el el bolsillo, simulando un teléfono móvil. En la elaboración de los datasets se ha utilizado la función 'Data Log' del dispositivo, dado que los datos se recogían a una frecuencia más elevada comparado con el otro método.

Los datasets creados se hicieron con el dispositivo en el bolsillo estando parado

y caminando. Cabe destacar el la función 'Data Log' guarda los datos en ficheros CSV y la aplicación que genera el árbol de decisiones solo admite ficheros de texto. Por lo tanto, se ha elaborado un pequeño código en python que se encarga de leer un fichero CSV y separar todos los valores por filas y columnas. De esta manera, guardar los datos necesarios en un fichero .txt y generar una cabecera es más efectivo. La cabecera que se pone al final en el fichero es para indicar las columnas a que tipos de datos pertenecen.

El núcleo de aprendizaje automático (MLC) está configurado para funcionar a 104 Hz, las características se extraen utilizando ventanas de 104 muestras, por lo que la salida del clasificador del árbol de decisiones se actualiza dos veces por segundo (104 Hz / 52 = 2 Hz).

Sólo se utilizan los datos del acelerómetro. La escala completa se establece en 8 g. Se calcula solo una característica, la varianza del acelerómetro. Los dos estados que va a ser capaz de detectar el núcleo de aprendizaje automático serán:

- 4 Parado
- 0 Caminando

Cada clase a clasificar por el árbol de decisión debe ser caracterizada por uno o más registros de datos.

AlgoBuilder es una herramienta software capaz de diseñar un flujo de procesamiento personalizado y construir el firmware para las placas con (núcleo—CPU) STM3 acopladas con las expansiones MEMS, o placas en formato de forma de kits de evaluación y desarrollo como SensorTile.box.

Se creará un proyecto de firmware utilizando AlgoBuilder, que puede ser configurado para utilizar STM32CubeIDE para generar el archivo binario, STM32CubeProgrammer para programar el dispositivo SensorTile.box, y Unicleo GUI para mostrar la salida en tiempo real. Para empezar, se configurará antes de nada la aplicación Al-

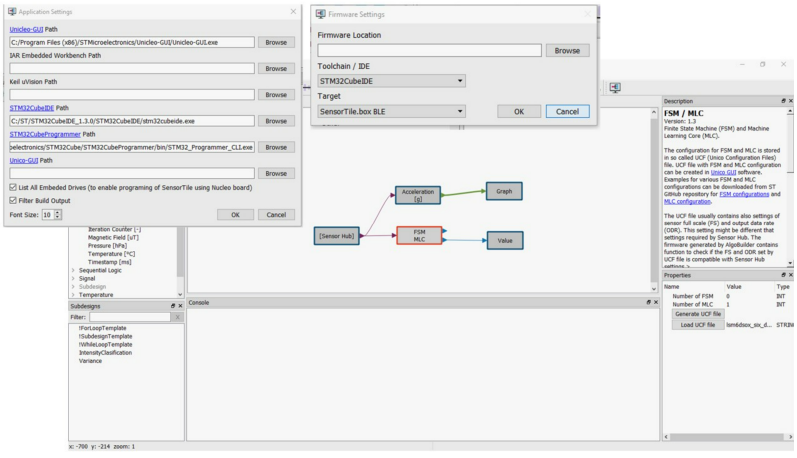
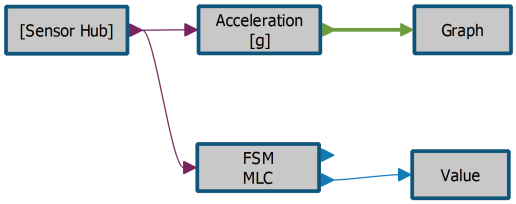


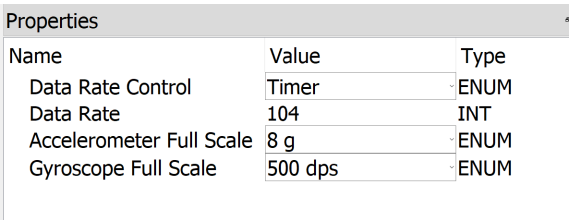
Figure 4.5: Configuracion de Algobuilder antes de generar el diseño del firmware.

goBuilder. Una vez abierta la aplicación, se le indicarán las rutas de Unicleo GUI, STM32CubeIDE y STM32CubeProgrammer. Estas aplicaciones son necesarias para el funcionamiento correcto de AlgoBuilder. Se creará un nuevo diseño indicando la ruta de guardado. Además, se indicará en el nuevo diseño que se creará para SensorTile.Box, como plataforma objetivo. Se configurará el bloque "Sensor Hub" por

defecto en el diseño para utilizar el temporizador como control de la tasa de datos, se establecerá la tasa de datos y la escala completa de los sensores de acuerdo con el ejemplo a cargar. Estos ajustes deben coincidir con la frecuencia a la que fueron recolectados los datos. A continuación, se puede expandir la librería 'Sensor Hub', arrastrar y soltar el bloque "FSM / MLC" en el diseño, conectarlo al bloque "Sensor Hub", haz clic y configurarlo para 0 Finite State Machine, 1 Decision Tree en el núcleo de Machine Learning.



(a) Esquema del diseño del firmware.



(b) Configuración de las propiedades del HUB

Opcionalmente, desde la biblioteca "Sensor Hub", se puede arrastrar y soltar el bloque "Aceleración", conectarlo al bloque "Sensor Hub". A continuación, desde la biblioteca "Pantalla", arrastrar y soltar el bloque "Gráfico", hacer clic y configúralo para el tipo de datos del acelerómetro, luego conectarlo al bloque Aceleración. Desde la biblioteca "Display", se puede arrastrar y soltar el bloque "Value", hacer clic y configurarlo para el tipo de datos personalizado, establecer el número de valores en 1, configurar los nombres de ventana, valor y unidad y, a continuación, conectarlo a la salida MLC del bloque de funciones FSM / MLC.

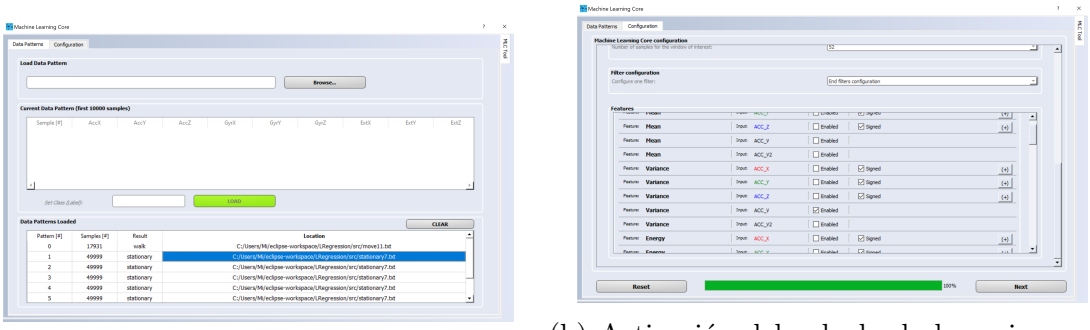
Para comenzar a cargar los datos y con la generación del árbol de decisiones, se va a seleccionar el bloque FSM/MLC. En las propiedades, al pulsar el botón 'Generate UCF file' se abrirá la aplicación de Unico GUI en modo offline. En esta aplicación nos dirigiremos a la herramienta 'Machine Learning Core Tool' y la abriremos. En la pestaña 'Data Patterns' para cada clase, se hará clic en 'Browse...' para seleccionar los archivos de registro de datos correspondientes (se pueden seleccionar varios archivos simultáneamente en el cuadro de diálogo), se escribirá la etiqueta de la clase y se hará clic en Cargar. Esta operación se repetirá por cada clase/estado de nuestro árbol de decisiones.

Una vez cargados todos los registros de datos, se puede seleccionar la pestaña 'Configuration' de la herramienta. En cada paso de la configuración se deberá de seleccionar la configuración expuesta a continuación y pulsar el botón 'Next'.

En este caso, los ajustes del acelerómetro son los mismos que los utilizados para la captura de datos (8 g, 104 Hz). El MLC está configurado para funcionar a la misma velocidad que el sensor del acelerómetro (104 Hz), y calcular un nuevo conjunto de características dos veces por segundo (se ha elegido una longitud de ventana de 52 muestras para obtener un buen equilibrio entre la latencia y el tiempo necesario para reconocer los movimientos).

Ajustes seleccionados en la pestaña Configuración:

1. LSM6DSOX sensor
2. MLC funcionando a 104Hz
3. Solo acelerómetro



(a) Carga de los ficheros de datos.

(b) Activación del calculo de la varianza de la aceleración

Figure 4.7: Configuración de Machine Learning Core

4. Acelerómetro configurado para una escala completa de 8g y una tasa de datos de 104Hz

5. Solo un árbol de decisión

6. Longitud de la ventana de 52 muestras

7. Sin filtro (seleccionar 'End filters configuration')

8. Seleccionar la varianza de la aceleración como valor a calcular

A continuación, al hacer click en 'Browse' se puede seleccionar la ruta de guardado y el nombre del archivo ARFF que se va a crear. Los archivos ARFF son archivos de texto: las primeras líneas describen los atributos (características extraídas), el resto del archivo tiene una línea por cada ventana (segmento de datos) de cada archivo de registro procesado por Unico. Cada línea consta de los atributos enumerados (un número o clase numérica) y la etiqueta correspondiente (una cadena, o clase nominal). Este archivo es la entrada para el algoritmo que diseña el clasificador de árbol de decisión.

Como último paso para la generación del árbol de decisión, hace falta introducir un código numérico asociado a cada etiqueta. Esta es la salida numérica del árbol de decisión cuando se ejecuta en el MLC. Se pueden crear grupos de un máximo de 4 etiquetas y luego utilizar la funcionalidad de metaclasificador del núcleo MLC. El primer grupo utiliza códigos numéricos de 0 a 3, el segundo grupo utiliza códigos de 4 a 7, el tercer grupo utiliza códigos de 8 a 11, y así sucesivamente. En este ejemplo, cada etiqueta está en un grupo diferente, por eso el código numérico es un múltiplo de 4.

Primero se le pedirá al usuario que especifique los metaclasificadores (lo cual no es necesario). Los meta-clasificadores pueden dejarse en 0.

Por último, se pedirá al usuario que seleccione el archivo UCF de destino y, al hacer clic en siguiente, éste será generado automáticamente por Unico.

UCF son las siglas de Unico Configuration File. Es un archivo de texto con una secuencia de direcciones de registro y valores correspondientes. Contiene la configuración completa del sensor, incluyendo por supuesto la configuración del MLC.

El archivo UCF puede ser utilizado tal cual por varias herramientas de software proporcionadas por ST: Unico GUI, Unicleo GUI, AlgoBuilder GUI.

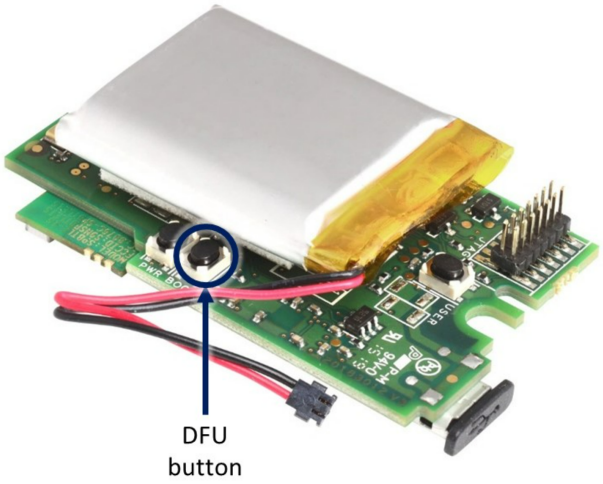


Figure 4.8: Activación del modo DFU del dispositivo

Los archivos UCF también pueden convertirse en código fuente C y guardarse como archivos .h de cabecera para incluirlos cómodamente en los proyectos C. El archivo UCF generado se cargará en el módulo FSM/MLC. Una vez cargado, se utilizará la aplicación AlgoBuilder para hacer el 'build' de la configuración firmware. Accediendo a la pestaña 'Firmware' y haciendo click a continuación en 'Build Firmware' conseguimos crear un archivo final que se debería cargar en el dispositivo.

Para cargar el firmware en el dispositivo, hace falta conectar SensorTile.Box al ordenador en modo DFU. El modo 'Direct Firmware Upgrade' es útil para hacer una actualización del dispositivo utilizando un cable con conexión micro-usb. Para activar el modo DFU, hace falta mantener pulsado el botón de 'boot' mientras se conecta el cable micro-usb del ordenador al dispositivo. Una vez conectado al ordenador, se puede soltar el botón 'boot'. A continuación, en AlgoBuilder, accedemos a la pestaña 'Tools' y pulsamos sobre el botón 'Program Target'. Esto activará el proceso de programación automática de nuestro dispositivo con el firmware creado.

Finalmente se puede comprobar el resultado conectando nuestro dispositivo programado al ordenador en el modo normal. Esto se consigue desconectando el dispositivo del ordenador (del modo DFU) y volviéndolo a conectar. Arrancamos la aplicación Unicleo-GUI, que inmediatamente detecta la conexión con el dispositivo SensorTile.Box. En la aplicación, desconectamos la memoria SD del dispositivo, esto permite poder ver los datos detectados por el dispositivo en la pantalla del ordenador. Pulsamos el botón 'Start' y abrimos el gráfico y el valor por pantalla. A continuación, si ponemos el dispositivo en el bolsillo y nos disponemos a caminar observaremos que el valor que marca la aplicación al caminar es de 4 al estar parado y 0 al caminar. También se puede apreciar en la gráfica del acelerómetro donde las señales varían más al caminar y son más estables al estar parado. Además, el núcleo de aprendizaje automático estuvo configurado para detectar con un valor numérico '0' el estado "caminando" y con un valor numérico '4' el estado "reposo". En las figuras que se muestran se puede apreciar que los resultados conseguidos por el bloque MLC coinciden con los valores detectados por el dispositivo en tiempo real. La varianza de los ejes del acelerómetro es mínima cuando se computa el estado de reposo en el bloque MLC y, por el contrario, la varianza del acelerómetro es máxima cuando el bloque BLC computa el estado de "caminar" con el valor cero.

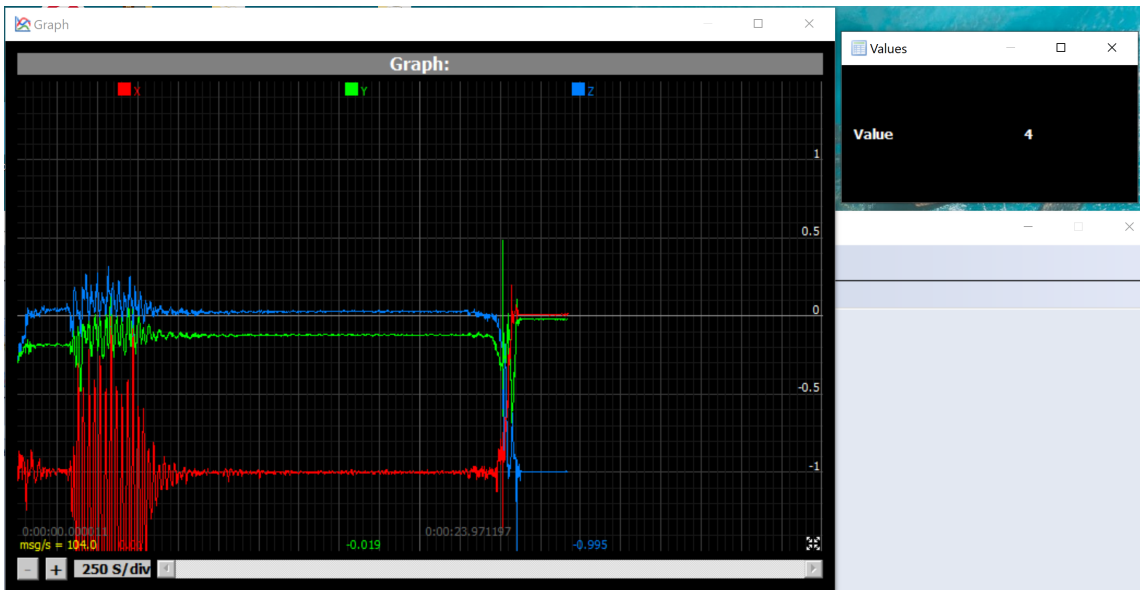


Figure 4.9: Muestra del estado 'parado'.

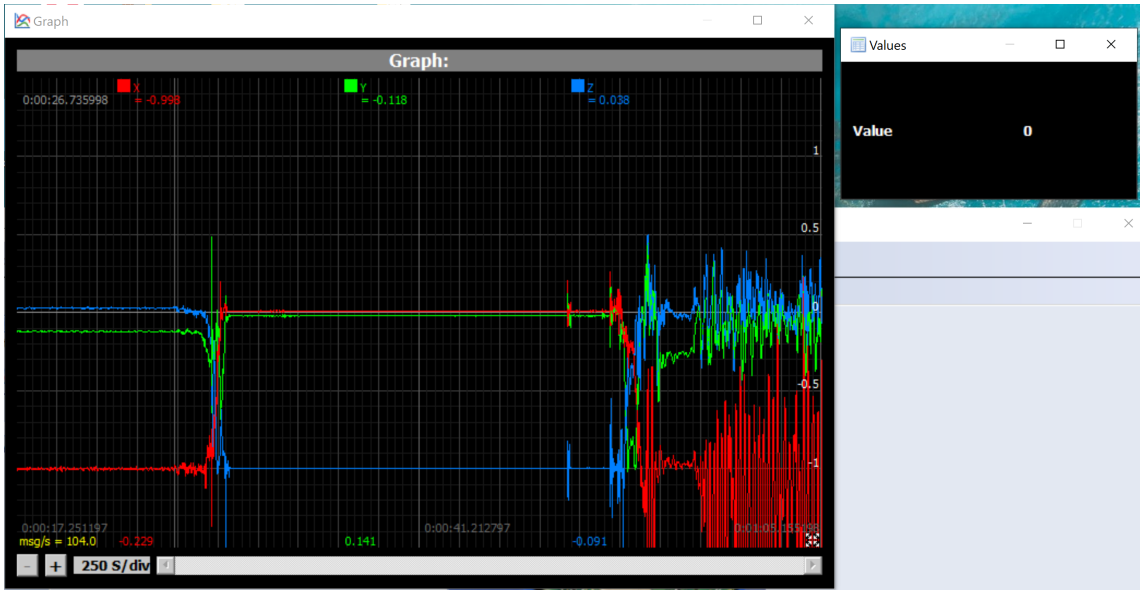


Figure 4.10: Muestra del estado 'caminando'.

Chapter 5

Conclusiones

Teniendo en cuenta los resultados obtenidos en este trabajo, considero que el dispositivo SensorTile.Box se debería de tener en consideración para la obtención y generación de datos en nuevas implementaciones automatizadas.

El problema que encuentro es la imperfección del firmware del dispositivo. Cada versión ofrecida del firmware por el fabricante depende de alguna manera de una conexión Bluetooth en todo momento. Efectivamente hay una versión del firmware que es capaz de recuperar la sesión una vez desconectes el dispositivo móvil. Sin embargo, esa versión no ofrece un control adecuado de la frecuencia de captación de los datos de los sensores. Cabe destacar que el dispositivo, según el hardware, debería de ser capaz de trabajar de forma autónoma sin la dependencia de una conexión Bluetooth, y por supuesto tener un control simultáneo más amplio de la frecuencia de captación de los datos.

En mi opinión, si se modificara el firmware para trabajar de manera autónoma, este dispositivo sería ideal para la captación de datos, automatización de procesos e implementación de nuevas aplicaciones mediante el uso de inteligencia artificial. A todo esto, cabe destacar que el dispositivo es bastante completo a la hora de hablar de los sensores de los que dispone. Eso lo convierte en un producto muy versátil.

En este trabajo queda demostrado el hecho que el dispositivo se puede entrenar para reconocer diversos estados analizando en tiempo real los valores de cada uno de sus sensores activos. El proceso necesario requiere captar los datos, clasificarlos para un entrenamiento posterior y configurar el dispositivo para el aprendizaje automático. Como resultado se obtiene un dispositivo que se podría implementar en sistemas más complejos o programar y reconfigurar para nuevas funcionalidades con un bajo coste. El hecho de que sea capaz de detectar las menores vibraciones de una superficie, sonidos y otros valores físicos ofrece una variedad amplia de posibles implementaciones.

También, al ofrecer la posibilidad de guardar los datos recogidos en una base de datos de la red, es posible utilizar estos datos y una potencia de procesado considerable para procesarlos y añadir valor a implementaciones futuras. Al reconfigurar el firmware de este dispositivo se podría utilizar para la detección de los animales domésticos, por ejemplo. Esto permitiría aumentar la calidad de producción de bienes de manera eficiente. Además, por otro lado se podría utilizar para analizar la calidad de conducción de las personas. Esto llegaría a ser algo muy relevante si tenemos en cuenta de que algunas empresas ya han empezado a analizar datos y entrenar máquinas para predicción de comportamiento de las personas.

Para llevar a cabo la implementación de aprendizaje automático del dispositivo se ha modificado el código firmware del dispositivo. Posteriormente se ha hecho una captura de datos. Los datos capturados se han etiquetado para poder ser reconocidos por la máquina de aprendizaje automático. Una vez obtenido el archivo de configuración de la inteligencia artificial del dispositivo, se ha comprobado que funciona según lo esperado. Además de recolectar los datos con la función 'Data Log', se ha configurado una alternativa más práctica que es la de guardar los datos en la red. De esta manera, si se dispone de varios dispositivos, se podría centralizar la información para posteriormente dedicarse al análisis de los datos obtenidos de manera efectiva. Esta segunda alternativa fue implementada para demostrar su uso y efectividad.

En resumen, el dispositivo ofrece ejemplos de firmware que todavía tienen errores. La conexión del dispositivo y la aplicación móvil tiende a ser inestable. Por el contrario, el usuario tiene la posibilidad de implementar un nuevo firmware con unas funcionalidades más robustas. Por ejemplo, proporcionar un factor de independencia de la conexión Bluetooth mediante la recuperación del estado de conexión previo con el dispositivo. Partiendo de esta idea, los futuros estudios e implementaciones con este dispositivo podrían enfocarse en la recolección y análisis de datos en adición al entonamiento y automatización de procesos en la industria.

Bibliography

[1] May 2021. URL: https://www.engineersgarage.com/what-are-inertial-sensors/.

[2] URL: https://www.mouser.at/new/stmicroelectronics/stm-steval-mksbox1v1-sensortile-box-kit/#Image-4.

[3] URL: https://www.apachefriends.org/es/about.html.

[4] *2.25 V low-voltage local digital temperature sensor*. URL: https://www.st.com/en/mems-and-sensors/stts751.html. (accessed: 22.07.2021).

[5] *3-axis accelerometer, ultra high resolution, low-noise, SPI 4-wire digital output, ±2.5g full-scale*. URL: https://www.st.com/en/mems-and-sensors/lis3dhh.html. (accessed: 22.07.2021).

[6] *3-axis MEMS accelerometer, ultra low power, configurable single/double-tap recognition, free-fall, wakeup, portrait/landscape, 6D/4D orientation detections*. URL: https://www.st.com/en/mems-and-sensors/lis2dw12.html. (accessed: 22.07.2021).

[7] Shivam Agarwal. "Data Mining: Data Mining Concepts and Techniques". In: (Dec. 2013), pp. 203–207. DOI: 10.1109/ICMIRA.2013.45.

[8] Jesús Álvarez et al. *BIENESTAR ANIMAL. Salud y enfermedad en relación con el comportamiento*. July 2017.

[9] *Apache HTTP Server Project*. URL: https://httpd.apache.org/. (accessed: 22.07.2021).

[10] Ahmad T. Azar and Shereen M. El-Metwally. "Decision tree classifiers for automated medical diagnosis". English. In: *Neural computing applications* 23.7 (2013), pp. 2387–2403.

[11] Meta Brown. *Transforming Unstructured Data into Useful Information*. Mar. 2014, pp. 211–230. ISBN: 978-1-4665-6870-9. DOI: 10.1201/b16666-11.

[12] *Capacitive digital sensor for relative humidity and temperature*. URL: https://www.st.com/en/mems-and-sensors/hts221.html. (accessed: 22.07.2021).

[13] *Changes in MySQL 8.0.26 (2021-07-20, General Availability)*. URL: https://dev.mysql.com/doc/relnotes/mysql/8.0/en/news-8-0-26.html. (accessed: 22.07.2021).

[14] Peter Cheeseman and John Stutz. "Bayesian Classification(AutoClass):Theory and Results". In: *Advances in Knowledge Discovery and Data Mining* (May 1997).

[15] Carlo Corsi. "Smart Sensors: Why and when the origin was and why and where the future will be". In: vol. 8993. Dec. 2013, p. 899302. DOI: 10.1117/12.2030025.

[16] *Different Types of Sensors/Detectors/Transducers*. URL: https://www.thomasnet.com/articles/instruments-controls/sensors/#level. (accessed: 22.07.2021).

[17] *Eclipse Mosquitto - An open source MQTT broker*. URL: https://mosquitto.org/. (accessed: 22.07.2021).

[18] *El Análisis De Cluster, Estadísticas, Kmeans La Agrupación Imagen Png*. URL: https://www.freepng.es/png-0wt4hs/. (accessed: 04.07.2021).

[19] *Esquema general del dispositivo y sus sensores*. URL: https://www.arrow.com/en/research-and-events/videos/getting-started-with-the-sensortilebox-iot-dev-kit.

[20] *High performance MEMS audio sensor single ended analog bottom-port microphone*. URL: https://www.st.com/en/mems-and-sensors/mp23abs1.html. (accessed: 22.07.2021).

[21] *High-performance MEMS nano pressure sensor: 260-1260 hPa absolute digital output barometer*. URL: https://www.st.com/en/mems-and-sensors/lps22hh.html. (accessed: 22.07.2021).

[22] *iNEMO inertial module with Machine Learning Core, Finite State Machine and advanced Digital Functions. Ultra-low power for battery operated IoT, Gaming, Wearable and Personal Electronics*. URL: https://www.st.com/en/mems-and-sensors/lsm6dsox.html. (accessed: 22.07.2021).

[23] *Introduction to Microcontrollers*. URL: https://www.arrow.com/en/research-and-events/articles/engineering-basics-what-is-a-microcontroller. (accessed: 22.07.2021).

[24] *LSM6DSOX: Machine Learning Core*. URL: https://www.st.com/resource/en/application_note/dm00563460-lsm6dsox-machine-learning-core-stmicroelectronics.pdf.

[25] *Magnetic sensor, digital output, 50 gauss magnetic field dynamic range, ultra-low power high performance 3-axis magnetometer*. URL: https://www.st.com/en/mems-and-sensors/lis2mdl.html. (accessed: 22.07.2021).

[26] *MQTT Protocol - Velotio*. URL: https://www.velotio.com/engineering-blog/mqtt-protocol-overview. (accessed: 20.07.2021).

[27] Xenia Naidenova. "Classification reasoning as a model of human commonsense reasoning". In: *CEUR Workshop Proceedings* 939 (Jan. 2012), pp. 57–64.

[28] *Proveedor de dispositivos electrónicos*. URL: www.mouser.com.

[29] Claude A. Pruneau. *Data Analysis Techniques for Physical Scientists*. Cambridge University Press, 2017. DOI: 10.1017/9781108241922.

[30] Raveena Rao. "Prime Faraday Technology Watch ISBN 1-84402-020-7 An Introduction to MEMS An Introduction to MEMS (Micro-electromechanical Systems) PRIME Faraday Partnership PRIME Faraday Partnership". In: (). URL: https://www.academia.edu/5077471/Prime_Faraday_Technology_Watch_ISBN_1_84402_020_7_An_Introduction_to_MEMS_An_Introduction_to_MEMS_Micro_electromechanical_Systems_PRIME_Faraday_Partnership_PRIME_Faraday_Partnership.

[31] Bal S Sandhu. "An Overview of MEMS Sensors". In: *Usa: Arm* (2015).

[32] T. Schroeder, S. Goddard, and B. Ramamurthy. "Scalable Web server clustering technologies". English. In: *IEEE network* 14.3 (2000), pp. 38–45.

[33] Belle Selene Xia and Peng Gong. "Review of business intelligence through data analysis". English. In: *Benchmarking : an international journal* 21.2 (2014), pp. 300–311.

[34] *SensorTile.box wireless multi sensor development kit with user friendly app for IoT and wearable sensor applications*. URL: https://www.st.com/en/evaluation-tools/steval-mksbox1v1.html. (accessed: 22.07.2021).

[35] *The Process of Data Discovery - CESSDA TRAINING*. URL: https://www.cessda.eu/Training/Training-Resources/Library/Data-Management-Expert-Guide/7.-Discover/The-process-of-data-discovery. (accessed: 18.07.2021).

[36] S. Velickov and D. Solomatine. "Predictive Data Mining : Practical Examples". In: (2000).

[37] *Very low power application processor module for Bluetooth® low energy v5.2*. URL: https://www.st.com/en/wireless-connectivity/bluenrg-m2.html. (accessed: 22.07.2021).